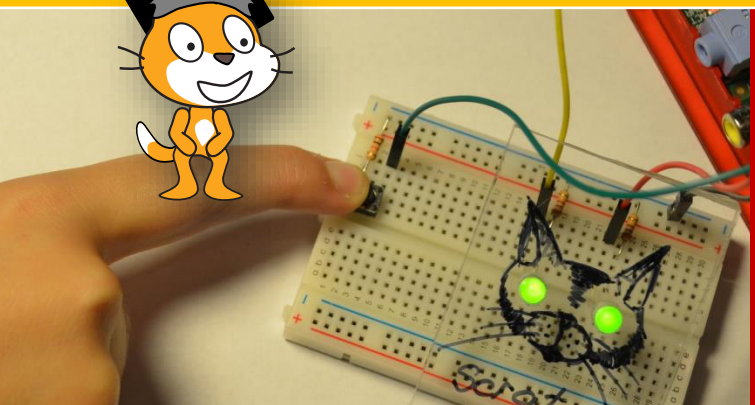
Raluca Elena MATEI
Mircea Tiberiu MATEI
Mara Sandra MATEI

Introducere în Scratch

Primii pași în programare



EUROGAMA INVENT

Introducere în Scratch

Ghid de programare pentru începători

Raluca Elena MATEI
Mircea Tiberiu MATEI
Mara Sandra MATEI

Tiparul executat la **S.C. Interbrand Impex S.R.L.**
Strada Oboga nr. 19 - 23, sector 6, cod poștal 062197, București
Tel.: +40 314 016 162 (63)
+40 722 279 226
Email: contact@digitalprints.ro

Anul tipăririi: 2015

Descrierea CIP a Bibliotecii Naționale a României

RALUCA ELENA, MATEI
Introducere în Scratch: Primii pași în programare / Raluca Elena MATEI, Mircea Tiberiu MATEI, Mara Sandra MATEI. – București : Editura EUROGAMA INVENT, 2015

ISBN 978-973-87690-9-0

Editura EUROGAMA INVENT

CUPRINS

Prefață

Povestea cărții

M-am născut și crescut in familia Matei. Acum doresc să-mi iau rămas bun pentru a pleca în lume să-mi croiesc propriul meu drum. Am și un scop: să-i învăț pe copii să programeze.

Am câteva capitole, nu prea multe. Dar nu-mi pare rău. Puteți să mă citiți pe sărite. Nu țin la etichetă sau bonton. Vă propun doar să-mi rezervați un loc în loja memoriei voastre. Să știți că exist, iar când veți simți că e nevoie să mă scărmănați printre file o să vă împărtășesc secretele mele.

Capitolul 1 ar trebui să vă convingă mai întâi părinții dacă merită să adaugați pisoiul Scratch în lista de prieteni.

Capitolul 2 e teorie. Dacă suferiți de insomnii, sigur o să fie cel mai bun medicament. Dacă vă e somn evitați-l.

Capitolul 3 vă prezintă lumea Scratch.

Capitolul 4 vă povestește despre blocurile vrăjite din Scratch, blocuri ce pot fi asamblate precum un puzzle pentru a obține programele.

Capitolul 5 ... păi ... ei bine ... ar trebui citit mai la urmă. După cum spune Scratchy, e pentru când le știți pe toate. Dacă reușiți să treceți de încercările puse la cale de Scratchy și prietenii săi, veți obține permisul de liberă trecere în Ținutul Vrăjit al Programării.

Capitolul 6 nu se recomandă fără ajutorul unui adult. Mai bine spus, vă trebuie puțin ajutor.

Capitolul 7 poate fi citit ... primul. Să fim sinceri nici nu-i chiar capitol .. sunt doar niște sfaturi.

Povestea Marei

Am descoperit minunata lume a programării Scratch cu ajutorul lui tati. La început, deși mă atrăgea pisoiașul portocaliu de pe ecran, mi se părea greu și plictisitor: de ce să fac jocuri, când sunt deja altele făcute?!? Acum știu răspunsul: este mult mai distractiv așa! Atunci nu îmi dădeam seama de asta, căci aveam doar șase ani. Strigam într-una:

- Nu vreeeaaaau!!! Nu vreau să învăț Scratch!!!

În gândul meu spuneam că orice s-ar întâmpla, nu voi învăța să programez.

Apoi, în vacanța de vară mi s-a schimbat părerea. Făceam împreună cu mami și cu prietena mea Teodora, progrămele drăguțe cu fețe zâmbitoare și triste, cu caracatițe și peștișori ce se plimbau pe ecran sau cu fantomițe ce nu-și găseau starea... și ... surpriză! A început să-mi placă programarea!!!

După ce am învățat bine să programez, câteva personaje parcă s-ar fi trezit la viață și m-au chemat în lumea lor. Acestea sunt prietenii mei: Scratchy – pisoiul portocaliu, Pată – cățelușul cu pată albastră, Codiță – maimuțica înfometată, Snowy – omul de zăpadă pe care l-am învățat să patineze.

Ei m-au condus prin toate cotloanele noului ținut descoperit, Ținutul Programării și mi-au spus multe din tainele Scratch-ului.

Și-am încălecat pe un led,

Și-am zis povestea mea (cred)!

Povestea lui mami…

Cum încep poveștile? Gata ... mi-am adus aminte...

A fost odată ca niciodată, o mămică și Fetița ei.

Mămica își dorea foarte mult să o învețe pe Fetiță să programeze. Însă Fetița era foarte mică și foarte îndărătnică.

Orice ar fi încercat mămica, cu vorbă bună sau păcăleli, nu reușea nimic din ceea ce își propunea. De fiecare dată când mămica îi vorbea Fetiței de comenzi date calculatorului, aceasta își strîngea mănuțele la piept și spunea: “Nu vreau!” ... “Nu e drept!!” ... “Nu vreau!!!”.

Și tot așa, până când, într-o bună zi, Fetița a făcut cunostință cu pisoiul Scratchy.

Însoțiți de Scratchy, mămica și Fetița au plecat împreună, într-o călătorie minunată, plină de surprize, spre Ținutul Vrăjit al Programării. Împreună au trecut prin multe încercări, iar Fetița și-a făcut mulți prieteni.

În cele din urmă Fetița a reușit să treacă testul final, rezolvând probleme cu care Scratchy, Snowy, Codiță și Pată au pus-o la încercare pentru a obține permisul de liberă trecere în Ținutul Programării.

Scratchy te invită și pe tine să îl urmezi pe acest drum fascinant. Nu o să îți pară rău!

Povestea lui tati….

Am descoperit Scratch dintr-o întâmplare. Eram în căutarea unei aplicații care să ofere un mediu de programare ce ar putea fi înțeles de un copil. Pe vremea aceea Mara, fiica noastră, avea șase ani. Printre "jucăriile" de acasă se afla și un minicalculator Raspberry PI cu care mai cochetam din când în când, făcând temerare incursiuni în tainele electronicii. Într-o sâmbătă, zi de toamnă cu multă ploaie, numai bună să stai ascuns în casă pentru a pune la dospit idei, ne-am așezat in jurul "focului" din ledurile lui PI. Cu un snop de senzori împrăștiat pe masa de lucru, privind în gol către ecranul monitorului, încercam să fac loc unui gând care să-mi aducă neliniștea acelei zile. Mara m-a readus în lumea reală cu o întrebare simplă:

- Tati, ce-i cu pisoiul acela pe ecran?

Ce am răspuns, o să mă întrebați. Sau, deja ați ghicit. Răspunsul este chiar această carte. Și-am încălecat pe o șa și v-am spus povestea așa.

Capitolul 1. ***Introducere***

Recomandat părinților. De evitat pentru persoanele sub 14 ani ☺!

Scratch este un limbaj de programare educaţional, realizat de "Grupul Lifelong Kindergarten" de la "Institutul de Tehnologie din Massachusetts". Cercetatori americani de la MIT au realizat un mediu ce permite crearea de aplicații grafice animate, jocuri video și aplicații interactive într-un mod simplu și intuitiv, fără a necesita cunoștinte de programare.

„*Copiii interacționează în ziua de azi cu tot felul de lucruri dinamice pe ecrane, dar aceasta este unidirecțională - ei folosesc lucruri pe care alții le creează*," a afirmat Mitchel Resnick de la MIT Media Lab, unul dintre autorii proiectului. „*Cu Scratch dorim sa le permitem copiilor ca ei sa fie creatorii. Dorim ca ei să creeze lucruri dinamice, interesante pe calculator.*"

Mediul este multilingvistic. Dacă este încorporată traducerea corespunzătoare, fiecare utilizator poate edita sursa în limba proprie, și mai mult, poate vizualiza sursa celorlalţi în propria limbă.

***Observație**: Puteți folosi mediul în limba română. Totuși, pentru că ne dorim să-i apropiem pe copii de expresiile consacrate din limba engleză, vom folosi și versiunea în limba engleză. Așa cum o să vedeți, forma și culoarea blocurilor de program este unică și independentă de limbă.*

Paralel cu dezvoltarea limbajului de programare Scratch, s-a dezvoltat şi site-ul Scratch (*http://scratch.mit.edu*), unde oricine poate încărca propriile „creaţii”. În acest fel, a apărut o comunitate, unde se poate cere ajutor şi suport, se pot exprima impresii despre munca altora, dar în acelaşi timp trebuie suportată părerea celorlalţi, chiar dacă poate conţine şi critici negative.

SCRATCH este un limbaj de programare care permite crearea propriilor povești interactive, animații, jocuri, muzică și artă. Scratch a fost dezvoltat cu scopul de a permite dezvoltarea creativității copiilor și de a le încuraja raționamentul logic.

1.1 Instalăm sau putem sări peste acest pas?

Răspunsul este depinde. Există disponibile două variante: o versiune desktop și alta online. Dacă avem conexiune la internet putem folosi varianta online. Dacă vrem să programăm oriunde, oricând și, în plus, să facem mici automatizări vom alege versiunea Desktop.

De menționat că proiectele pe care le realizăm pot fi utilizate în oricare dintre variante.

1.1.1 Versiunea desktop

Se poate descarca de la adresa *http://scratch.mit.edu/scratch_1.4/*.

Sunt oferite variante pentru următoarele sisteme de operare:

- Windows - 2000, XP, Vista, 7 și 8
- Mac OS X - versiunea 10.4 sau mai nouă
- Linux – Ubuntu 12.04 sau mai nou

Cerințele hardware sunt următoarele:

- Display minim 800x480, 16-bit
- Disk minim 120 MB

- Memorie și CPU – fără cerințe speciale, majoritatea calculatoarelor de generație recentă vor rula fără probleme
- Placa de sunet cu boxe și microfon, dacă dorim aplicații media
- Cameră video – există suport pentru camere USB sau camere încorporate

1.1.2 Versiunea online

Nu necesită instalare și poate fi accesată din orice browser web la adresa *http://scratch.mit.edu/* .

1.2 Elemente de bază ale unui proiect

Un proiect Scratch este construit cu obiecte numite „personaje” (sprites).

Modul de prezentare a unui personaj poate fi personalizat cu ajutorul costumelor. Cu ajutorul costumelor un personaj poate să arate ca o persoană, un animal, un obiect sau orice altceva.

Costumele pot fi create cu ajutorul utilitarului **EditorPaint**, pot fi alese dintr-o colecție predefinită sau pot fi importate de pe un site web. Costumul implicit este **Scratch**, o pisicuță.

Prin intermediul instrucțiunilor, un obiect se va putea mișca, va putea reda muzica și va putea relaționa cu alte personaje.

Realizarea unui program (script) în Scratch se bazează pe îmbinarea unor fragmente de cod – numite blocuri. Aceste blocuri se încadrează în următoarele categorii: Motion, Looks, Sound, Pen, Data, Events, Control, Sensing, Operators, Broadcast și More Blocks (blocuri construite la comandă).

Blocurile și ordinea lor sunt foarte importante, deoarece acestea determină modul în care personajele interacționează unele cu altele.

La script-uri pot fi atașate comentarii pentru a explica ceea ce fac anumite blocuri și în ce scop a fost realizat script-ul.

Fiecare bloc este proiectat astfel încât:

- să determine rularea un script:

Fig. 1.1 Exemplu de bloc ce determină rularea unui script

- să fie adăugat la sfârșitul unui script:

Fig. 1.2 Exemplu de bloc ce poate fi adăugat la sfârșitul unui script

- să semnalizeze sfârșitul unui script:

Fig. 1.3 Exemplu de bloc ce semnalizează sfâșitul unui script

- să fie potrivit în interiorul altor blocuri:

Fig. 1.4 Exemplu de bloc ce poate fi potrivit în interiorul altor blocuri

- să conțină și alte blocuri.

Fig. 1.5 Exemplu de bloc ce poate să conțină și alte blocuri

Din acest motiv, pentru a scrie un program, blocuri vor fi asamblate precum un joc de puzzle. Acest fapt previne erori de sintaxă.

Capitolul 2. **Algoritmii**

Știu. E plictisitor. Iar teorie!

Dacă sunteți nerăbdător puteți sări la următorul capitol. Dar, nu uitați să vă întoarceți. De ce? O să vă răspund tot cu o întrebare. V-ar place să jucați un joc, fără să cunoașteți regulile care vă transformă în câștigător?

Când se scrie codul unui program, practic se comunică calculatorului pașii pe care trebuie să-i urmeze pentru a ajunge la un anumit rezultat. Această succesiune de pași poartă numele de algoritm.

Algoritmul este descris cu ajutorul instrucțiunilor și sintaxelor specifice fiecărui limbaj de programare.

2.1 Obiectele cu care lucrează algoritmii

Datele cu care lucrează algoritmii se pot clasifica după următoarele criterii:

- în funcție de posibilitatea de a-și modifica valoarea avem constante și variabile;
- după tipul datelor: numerice (naturale, întregi, reale), logice (booleene), alfabetice;
- organizarea datelor: simple, structurate.

Constantele sunt valori ce nu se modifică în cadrul programului.

Variabilele sunt date ale caror valori se pot modifica la fiecare execuție a programului sau chiar în timpul unei execuții.

Expresiile sunt utilizate în scopul efectuării calculelor. O expresie este alcătuită din unul sau mai mulți operanzi legați între ei prin operatori.

Operanzii pot fi constante, variabile sau expresii încadrate între paranteze rotunde.

Operatorii desemnează operații care se execută asupra operanzilor.

Evaluarea unei expresii presupune înlocuirea variabilelor din expresie cu valorile lor curente și efectuarea calculelor. În procesul de evaluare a unei expresii se respectă regulile de bază învățate la matematică (se ține cont de prioritatea operatorilor și de asociativitatea acestora).

În urma evaluării unei expresii se obține o valoare rezultat, tipul acestei valori indicând tipul expresiei.

Prin identificator înțelegem un nume asociat unei constante, variabile, funcții, etc. Un identificator poate conține numai litere, cifre și caracterul special "_", primul caracter fiind obligatoriu o literă mare sau mică sau caracterul "-".

Cuvintele cheie ale limbajului sunt identificatori cu semnificație fixată, care nu pot fi folosiți în alt context decât cel precizat în definirea limbajului.

2.2 Structuri de control

Structurile de control permit controlul desfăşurării programului.

2.2.1 Structuri simple

2.2.1.1 Structura de atribuire

Atribuirea reprezintă operația prin care unei variabile i se atribuie o valoare.

Reprezentarea în schemă logică:

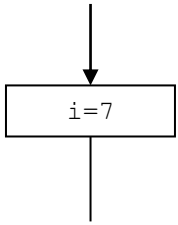


Fig. 2.1 Atribuire

2.2.1.2 Structura de intrare/ieșire

Operațiile de intrare/ieșire sunt acelea cu ajutorul cărora programul ia de la tastatură o valoare (intrarea), respectiv afișează pe ecran o valoare (ieșirea).

Reprezentarea în schemă logică:

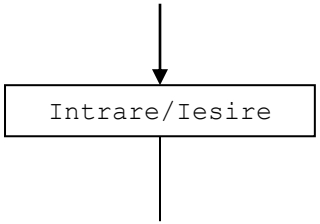


Fig. 2.2 Intrare/Ieșire

2.2.1.3 Condiția

Condiția reprezintă practic o intrebare ridicată de programator într-un moment în program. În funcție de răspunsul la întrebare - care poate fi ori “Da”, ori “Nu” - programul se continuă pe una din ramuri.

Reprezentarea în schemă logică:

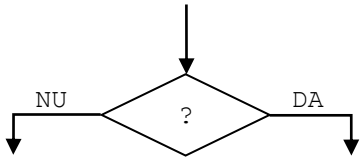


Fig. 2.3 Condiție

2.2.2 **Structuri decizionale**

2.2.2.1 Structura alternativă

Această structură este deseori confundată de programatorii începători cu structura simplă condiție. Ea este însă alcatuită dintr-o condiție ***plus*** instrucțiunile care se execută dacă respectiva condiție este adevarată, respectiv instrucțiunile care se execută dacă este falsă.

Reprezentarea în schemă logică:

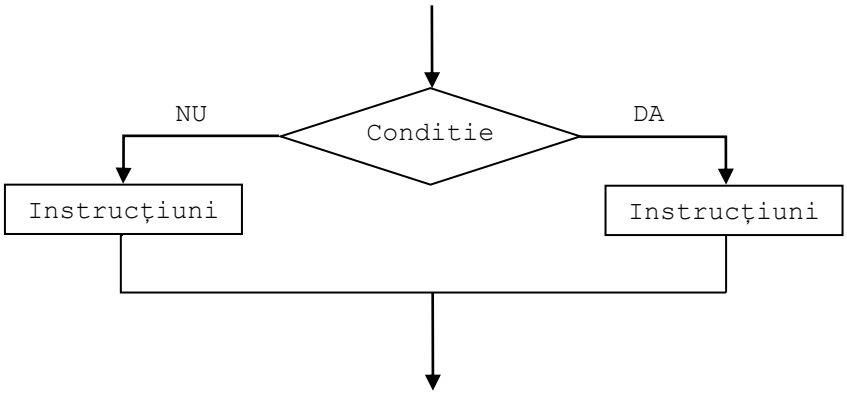


Fig. 2.4 Structură alternativă

2.2.2.2 Structura de selecție

În cazul în care o variabilă poate lua una dintre 5, 6, 7 sau chiar mai multe variante de valori, folosirea succesivă a structurilor alternative duce la îngreunarea lizibilității programului. În loc de 6 *If*-uri pentru 7 variante de valori, folosim structura de selecție.

Reprezentarea în schemă logică:

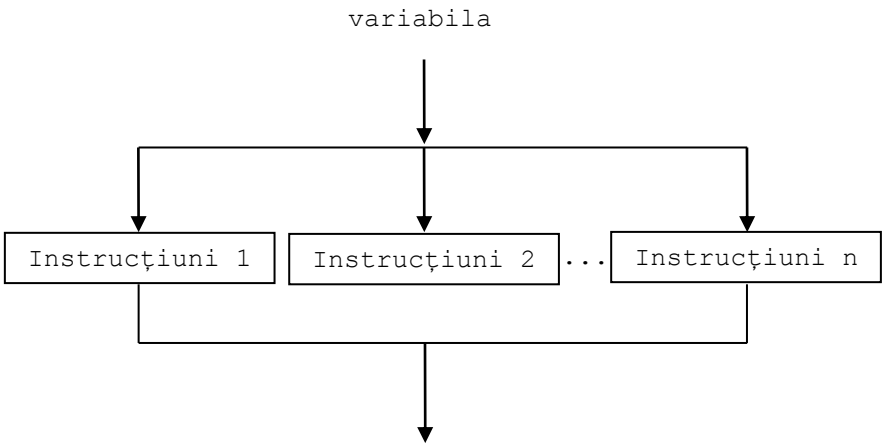


Fig. 2.5 Structură de selecție

2.2.3 **Structuri repetitive**

2.2.3.1 Structura repetitivă cu condiție inițială

Această structură este alcătuită dintr-o condiție, care se află la început, și un bloc de instrucțiuni, care se execută dacă rezultatul evaluării condiției este adevărat.

Reprezentarea în schemă logică:

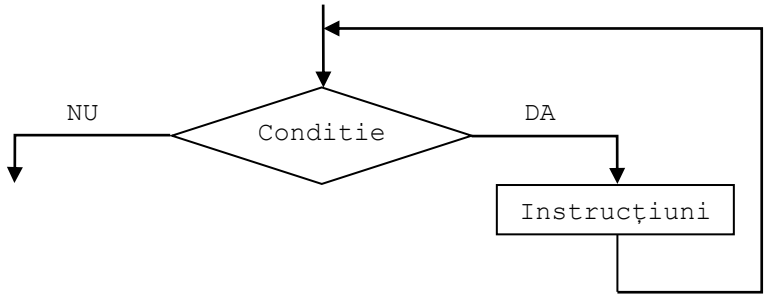


Fig. 2.6 Structură repetitivă cu condiție inițială

2.2.3.2 Structuri repetitive cu condiție finală

Alcătuirea ei, cum detectăm încă din nume, este de forma bloc de instrucțiuni, apoi condiție.

De remarcat că blocul de instrucțiuni se execută minim o dată, spre deosebire de structura repetitivă cu test inițial, unde

blocul de instrucțiuni era posibil să nu se execute deloc, dacă rezultatul evaluării condiției inițiale era fals.

Reprezentarea în schemă logică:

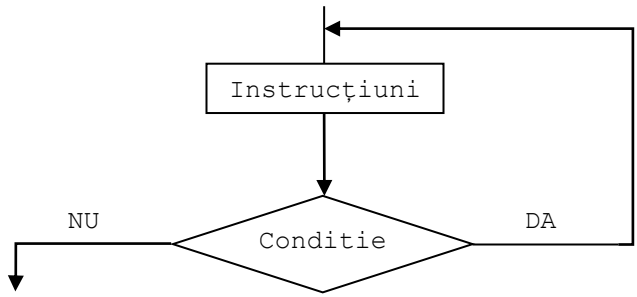


Fig. 2.7 Structură repetitivă cu condiție finală

2.2.3.3 Structura repetitivă cu contor

Este un caz particular al structurii de control cu test inițial. Utilizează o variabilă pe care o folosește ca un contor. Această variabilă are trei caracteristici:

- pleacă de la o valoare;
- ajunge la o valoare;
- înaintează cu un pas.

Reprezentarea în schemă logică:

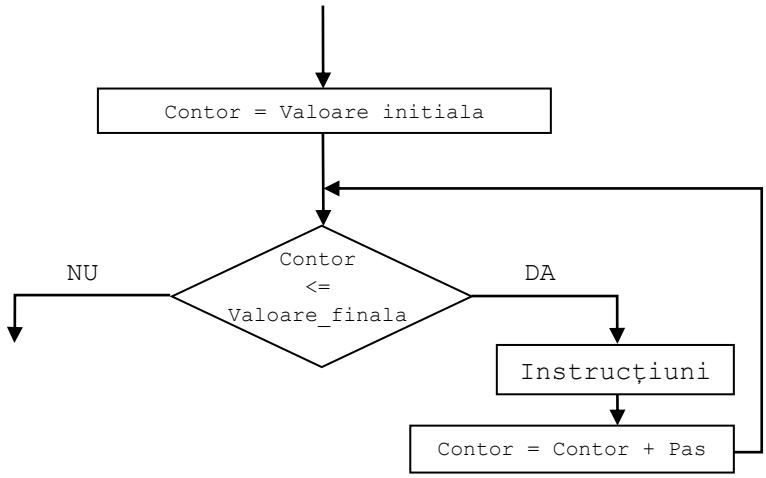


Fig. 2.8 Structură repetitivă cu contor

Capitolul 3. Interfața grafică Scratch

Hmmm. Aici e de citit. O singură dată e suficient. Garantez!

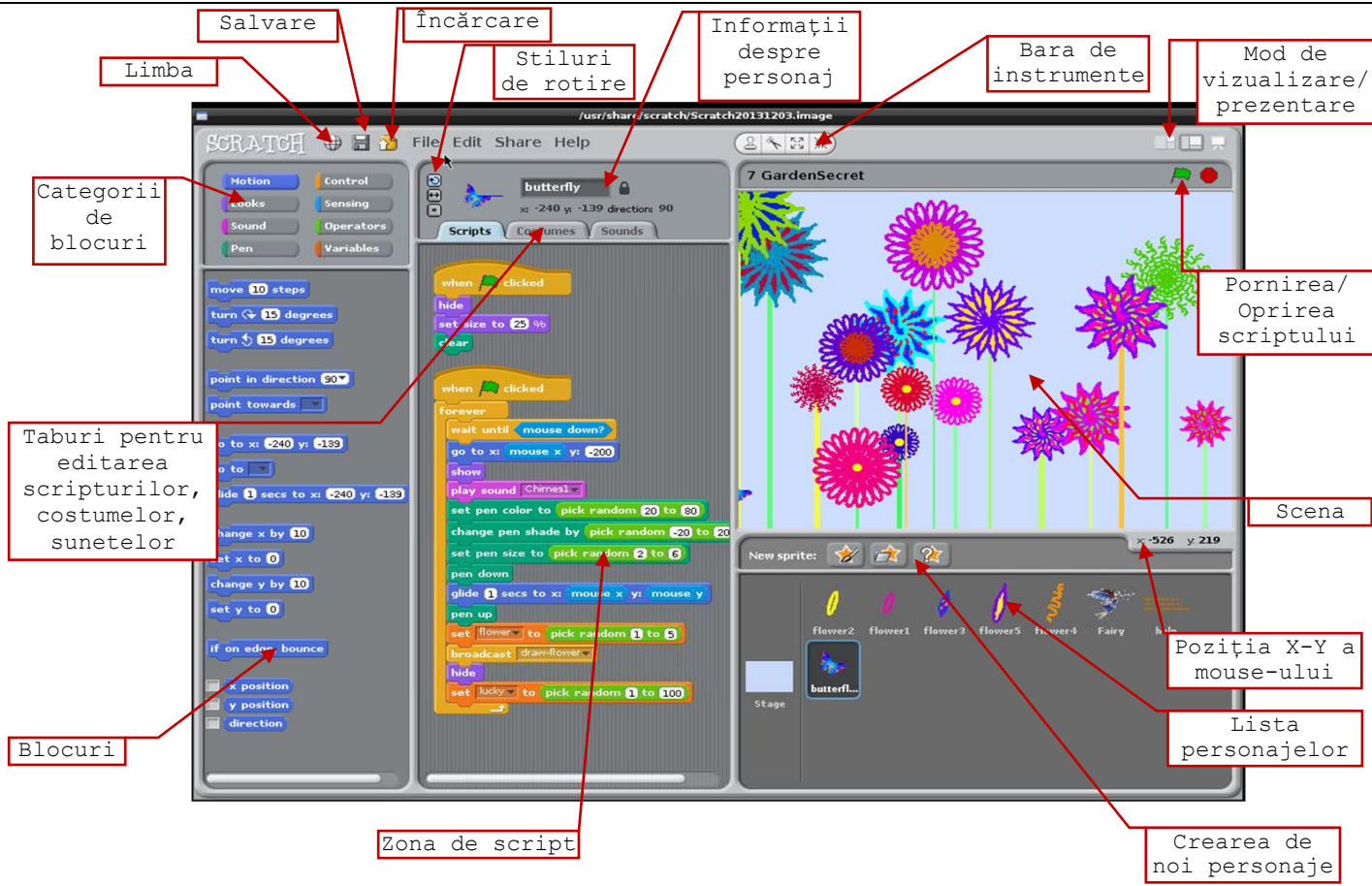
La proiectarea limbajului de programare Scratch, prioritatea realizatorilor a fost de a crea un mediu de dezvoltare pentru programare care să fie intuitiv şi care să poată fi uşor înţeles şi învăţat de copiii care nu au nicio experienţă anterioară în programare.

Interfaţa limbajului Scrath este prietenoasă, lizibilă, cu acces direct la comenzi. Utilizarea lor adecvată este sprijinită de fereastra Ajutor, care oferă asistenţă rapidă. Limba interfaţei se poate seta în funcţie de opţiunea programatorului. Astfel, utilizatorii care folosesc variante diferite pentru programare pot înţelege cu uşurinţă codul sursă a celorlalţi. Conţine o bară de meniu, bară de instrumente, interfaţă de programare.

Interfaţa de programare are trei părţi principale: secţiunea pentru comenzi, secţiunea de program şi secţiunea de joc.

Comenzile sunt afişate folosind structuri grafice pe care le distingem în funcţie de culoarea şi forma lor.

Deoarece comenzile nu trebuie tastate (se va folosi tehnica „drag-and-drop”), nu vor apărea greşeli de sintaxă în program. Rezultatul este imediat, nu sunt mesaje de eroare, astfel experienţa neplăcută a eșecului lipseşte. Executarea instrucţiunilor se poate urmări în secţiunea de joc, astfel sentimentul reuşitei este imediat. Deoarece codul sursă al programului şi rezultatul rulării programului sunt vizibile deodată, erorile logice se pot detecta şi, implicit, corecta rapid.


Salvare
Încărcare
Limba
Stiluri de rotire
Informații despre personaj
Bara de instrumente
Mod de vizualizare/ prezentare
Categorii de blocuri
Pornirea/ Oprirea scriptului
Taburi pentru editarea scripturilor, costumelor, sunetelor
Scena
Poziția X-Y a mouse-ului
Blocuri
Lista personajelor
Zona de script
Crearea de noi personaje
SCRATCH
File Edit Share Help
Motion
Control
Looks
Sensing
Sound
Operators
Pen
Variables
Scripts
Costumes
Sounds
7 GardenSecret
New sprite:
Stage

Secţiunea de program pe lângă sarcini (Scripts), conţine caracteristicile de costum (Costumes) şi de sunet (Sounds) pentru personaje.

3.1 Scena

Personajele se mișcă și interacționează unul cu altul pe scenă. Scena (stage) este fundalul proiectului, locul în care animațiile și jocurile prind viața.

Scena este un personaj cu anumite caracteristici speciale: nu se poate mișca, toate personajele pot fi mutate pe scenă, dar niciun personaj nu poate fi mutat în spatele scenei.

Scena are două butoane  . Green Flag (Steagul verde): permite rularea proiectului. Stop Sign (Bulina Rosie): oprește proiectul.

La fel cum un personaj își poate schimba aspectul prin comutarea costumelor și scena poate fi schimbată. Pentru a vedea și edita script-uri, fundaluri și sunete asociate cu scena, se dă clic pe pictograma corespunzătoare scenei.

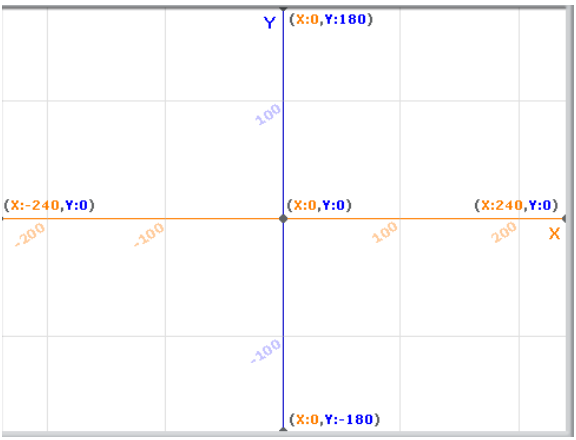


Fig. 3.1 Scena

Scena este de 480 de unități lățime și 360 de unități de înaltime. Aceasta este împărțită într-o grilă x-y. Fiecare punct al scenei are două coordonate: un x și un y.

Pentru a afla coordonatele xy, se va mișca cursorul mouse-ului și valorile curente pentru x și y vor fi afișate într-o zonă specială aflată chiar sub scenă.

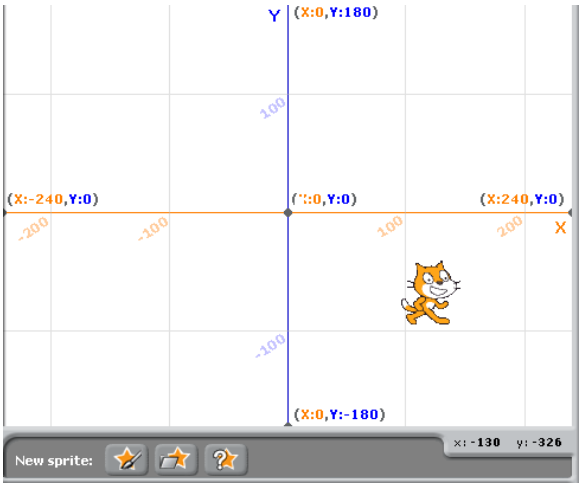


Fig. 3.2 Poziționarea unui personaj în cadrul scenei

3.2 Modul de prezentare


Pentru prezentarea proiectului se dă clic pe butonul "Mod de prezentare". Pentru a ieși din modul de prezentare apasă tasta Esc.


Butoanele View Mode se folosesc pentru a comuta între modul de vizualizare - scena mică/ mare.

3.3 Crearea unui personaj nou

La inițierea unui proiect există un singur personaj: pisica Scratch.

Pentru adăugarea unui personaj se vor utiliza butoanele: 

.

 Creează propriul costum pentru un nou personaj, utilizând utilitarul Paint Editor.

 Importă, la alegerea programatorului, un personaj sau un costum dintr-o colecție predefinită.

 Importă, aleator, un personaj dintr-o colecție predefinită.

Ștergerea unui personaj se poate realiza selectând comanda **delete** din meniul contextual ce apare atunci când facem clic-dreapta pe acesta.

Interfața mediului Scratch permite vizualizarea caracteristicilor personajului selectat: numele, poziția x-y, direcție, etc.

Fig. 3.3 Vizualizarea caracteristicilor personajului selectat

În această zonă este posibilă tastarea unui nume nou pentru personaj.

Linia albastră de pe miniatură arată direcția de mișcare a personajului (0 = în sus, 90 = dreapta, 180 = jos, -90 = stânga). Cu ajutorul acestei linii se poate schimba direcția personajului.

3.4 Lista personajelor

Lista tuturor personajelor din proiect apare sub forma unor miniaturi afișate într-o zonă aflată sub scenă. Numele personajelor

apare inscripționat sub fiecare miniatură. Imaginilor pot fi rearanjate prin glisarea acestora în cadrul listei.

Pentru editarea unui personaj (costume, fundal muzica) sau a scriptului asociat acestuia, facem clic pe miniatura din listă sau dublu-clic pe imaginea personajului aflat pe scenă. Personajul selectat este evidențiat printr-un contur albastru.

Fig. 3.4 Selectarea unui personaj

Pentru a afișa, exporta, copia sau șterge un personaj, facem clic dreapta pe miniatura corespunzătoare și se alege opțiunea dorită din meniul contextual.

3.5 Blocurile și zona de program (script)

Pentru a programa un personaj, blocurile se glisează în zona Script. Pentru a rula un bloc, se dă clic pe el.

Când un bloc este glisat în zona Script, un punct alb indică unde poate fi plasat blocul pentru a forma o conexiune validă cu un alt bloc.

Unele blocuri au câmpuri albe al căror text poate fi editat, cum ar fi 

.

Pentru a modifica valoarea, se dă clic în interiorul casetei albe și se tastează un număr sau se aduce un bloc cu colțuri rotunjite (

) în zona de text.

Alte blocuri au de asemenea meniuri derulante 

Clic pentru a vedea meniul, apoi clic din nou pentru a face o selecție.

3.6 Costume

Pentru a vedea sau edita costumele unui personaj, se selectează opțiunea Costumes (Fig.3.5).

Fig.3.5 Editarea costumelor unui personaj

Fig.3.6 Editarea fundalului sonor

Pentru a comuta la un costum diferit se dă clic pe imaginea redusă a costumului dorit. Fiecare costum are asociat un număr (afișat în stânga acesteia). Ordinea costumelor poate fi modificată prin glisarea miniaturilor.

Există o colecție de costume predefinite. În proiecte pot fi folosite costumele implicite sau pot fi create costume noi.

Există mai multe posibilități de a crea costume:

Clic pe butonul 

pentru a crea un costum cu ajutorul utilitarului Paint Editor.

Clic pe butonul 

pentru a importa un fișier imagine de pe hard disk.

Clic pe butonul 

pentru a realiza fotografii cu ajutorul unui webcam.

Scratch recunoaște mai multe formate de imagine: JPG, BMP, PNG, GIF (inclusiv GIF animat).

3.7 Sunete

Pentru a edita fundalul sonor corespunzator unui personaj se selectează opțiunea Sound. (Fig.3.6)

Într-un proiect pot fi utilizate sunete predefinite sau pot fi importate sau create noi sunete. Sunt recunoscute și pot fi prelucrate fișiere de tip MP3 și WAV necomprimate.

3.8 Meniul și bara de instrumente

Fig. 3.7 Meniul și bara de instrumente

3.8.1 Bara de instrumente

Bara de instrumente din partea de sus a programului conține multe funcții foarte importante în Scratch.

- Imagine glob  : apăsărea aceasteia oferă o listă verticală, cu toate limbile în care este disponibil programul.
- Imaginea dischetă  : apăsărea aceasteia permite salvarea proiectului.
- Imaginea dosar  : apăsărea aceasteia permite încărcarea proiectului în spațiul virtual Scratch (website).

❖ Imaginea ștampilă  : apăsărea acesteia permite copierea costumelor, sunetelor, blocurilor și scripturilor.

❖ Imaginea foarfecă  : apăsărea acesteia permite decuparea costumelor, sunetelor, blocurilor și scripturilor.

❖ Imaginea  : apăsărea acesteia duce la creșterea dimensiunii personajului.

❖ Imaginea  : apăsărea acesteia duce la micșorarea dimensiunii personajului.

3.8.2 **Opțiuni de meniu**

Prin intermediul meniului File se poate crea un proiect nou, se poate deschide un proiect existent și se poate salva proiectul în folderul Proiecte sau în alte locații.

Meniul Edit oferă mai multe caracteristici de editare pentru proiectul curent.

Prin intermediul meniului Share proiectul se poate încărca pe site-ul Scratch.

Din meniul Help se accesa o pagină de ajutor și multe link-uri utile.

3.8.3 **Limba română**

Pentru limba română, selectați pictograma glob, derulați până la limba română, conform figurii.

Surpriză: În lista limbilor există și limba miau (Meow)!

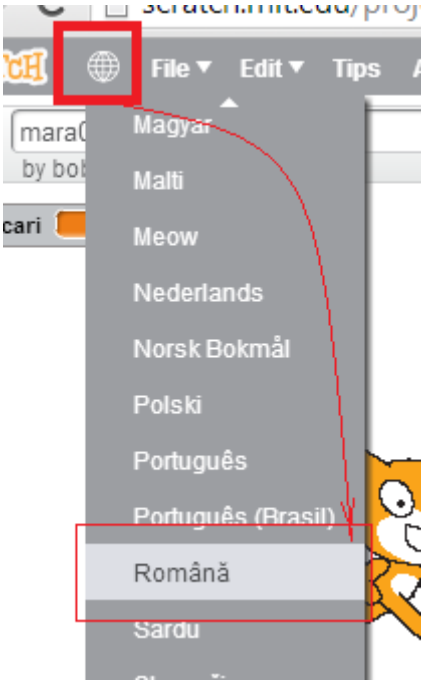


Fig. 3.8 Limba română

Inițial, pentru această carte am folosit opțiunea de limbă engleză. Scopul urmărit era de a apropia începătorii în tainele programării de termenii cu care o să se întâlnească în majoritatea limbajelor de programare. Însă, experiența mi-a arătat, că este necesar, pentru cei micuți, care abia încep să citească, să existe și varianta în limba română.

Capitolul 4. **Blocuri**

Nu sunt blocuri în care să locuiți. Sunt blocuri de piatră magică cu care puteți să vă creați o lume imaginară.

4.1 Blocurile EVENTS (Evenimente)

Blocurile de tip eveniment reprezintă una dintre cele zece categorii de blocuri Scratch. Acestea sunt folosite pentru a detecta evenimente ce determină rularea programelor.

When Green Flag Clicked – Rulează programul când se apasă steagul verde

Fig. 4.1 When Green Flag Cliced

When () Key Pressed - Rulează programul când se apasă tasta specificată

Fig. 4.2 When () Key Pressed

When This Sprite Clicked - Rulează programul când se dă clic pe personaj

Fig. 4.3 When This Sprite Clicked

When Backdrop Switches to () – Rulează programul când se schimbă scena

Fig. 4.4 When Backdrop Switches to ()

When () > () - Rulează atunci când atributul selectat (intensitate, timp, mișcare video) este mai mare decât o valoare specificată

Fig. 4.5 When () > ()

Broadcast (mesaj) – Se trimite un mesaj tuturor personajelor. Mesajul va fi recepționat și de scenă.

Fig. 4.6 Broadcast (mesaj)

Acest bloc este util atunci când dorim să punem personajele să execute ceva. Se folosește în pereche cu următorul bloc.

When I Receive () – Când se primește mesajul specificat, se execută o anumită secvență de program.

Fig. 4.7 When I Receive ()

4.2 Blocurile CONTROL

Aceste blocuri realizează controlul desfăşurării programului.

Wait (număr) Secs – Așteaptă un anumit număr de secunde, apoi continuă cu blocul următor.

Fig. 4.8 Wait (număr) Secs

Wait Until (Conditie) – Așteaptă până când condiția devine adevarată, apoi continuă cu blocul următor.

Fig. 4.9 Wait Until (Conditie)

Repeat (număr) – Execută blocurile din interior de un anumit număr de ori. Este un bloc corespunzător structurii repetitive cu contor.

Fig. 4.10 Repeat (număr)

Repeat Until (Conditie) - Testează dacă este falsă condiția; dacă da, ruleaza blocurile din interior și verifică din nou condiția. Dacă este adevarată, sare la blocurile care urmează. Este o structură de control repetitivă deoarece blocurile din interior sunt executate, în mod repetat, până când condiția devine adevarată.

Fig. 4.11 Repeat Until (Conditie)

Forever – Repetă, fără oprire, execuția blocurilor din interior. Corespunde structurii repetitive cu test inițial, în care condiția este mereu adevărată.

Fig. 4.12 Forever

If (Conditie) Then - Este un bloc corespunzător structurii de decizie. În cazul în care condiția este adevarată, se execută blocurile din interiorul zonei ***if.***

Fig. 4.13 If (Conditie) Then

If (Conditie) Then, Else – Este un bloc corespunzător structurii de decizie. În cazul în care condiția este adevarată, se execută blocurile din interiorul zonei ***if***; dacă nu, se execută blocurile din interiorul zonei ***else***

Fig. 4.14 If (Conditie) Then, Else

Stop () – Realizează același lucru ca și atunci când se dă clic pe butonul stop din partea de sus a ecranului.

Fig. 4.15 Stop ()

4.3 Blocurile MOTION (Mișcare)

Aceste blocuri sunt folosite pentru a controla mișcarea unui personaj în cadrul scenei.

Move (număr) Steps - permite mișcarea personajului cu numărul de pași specificat. Se poate schimba direcția de mers a personajului prin tastarea unui număr negativ (cum ar fi -10).

Exemplu

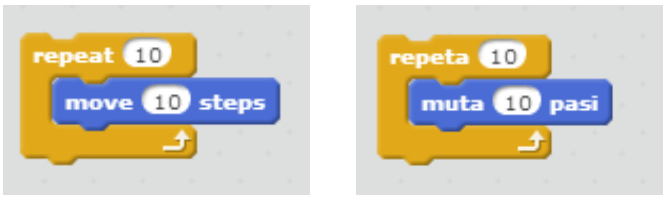


Fig. 4.16 Exemplu. Prima animație.

Exemplu

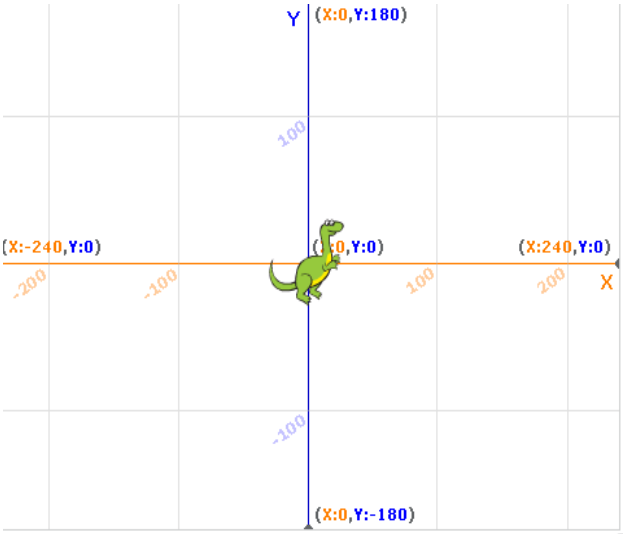


Fig. 4.17Inițial

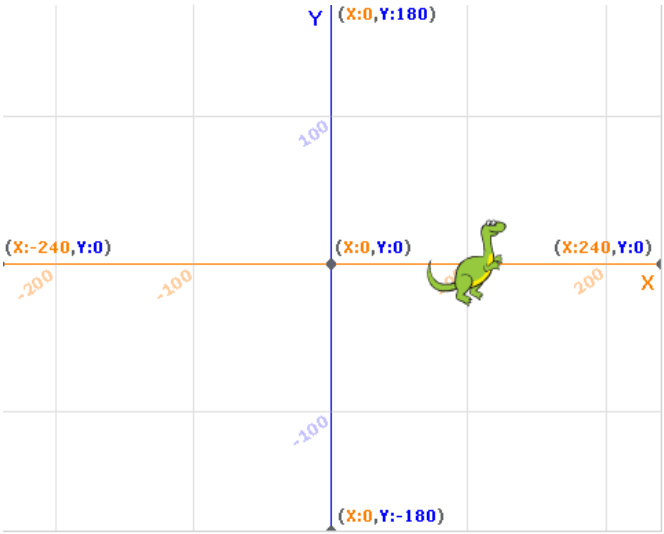


Fig. 4.18 Rezultat dupa rularea scriptului prezentat în Fig. 4.16

Fig. 4.19 Exemplu. Personajul va urmări pointerul mouse-ului

Turn Clockwise (număr) Degrees – Rotește personajul spre dreapta cu numărul de grade specificat.

Turn Counter-Clockwise () Degrees – Rotește personajul spre stânga cu numărul de grade specificat.

Exemple de utilizare a acestor blocuri sunt prezentate în Fig. 4.20 și în Fig. 4.21.

Fig. 4.20 Exemplu. O rotație simplă

Fig. 4.21 Exemplu. Animație

Point în Direction () – Setează direcția personajului.

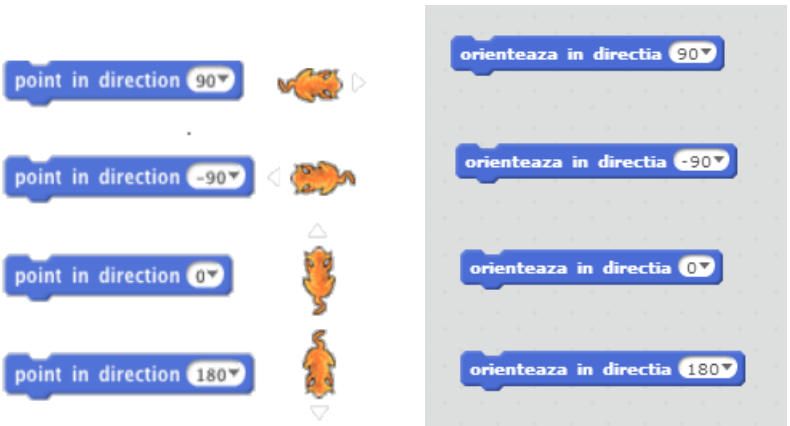


Fig. 4.22 Point în Direction ()

Point Towards () – Direcționează personajul spre cursorul mouse-ului sau spre un alt personaj.

Exemplu

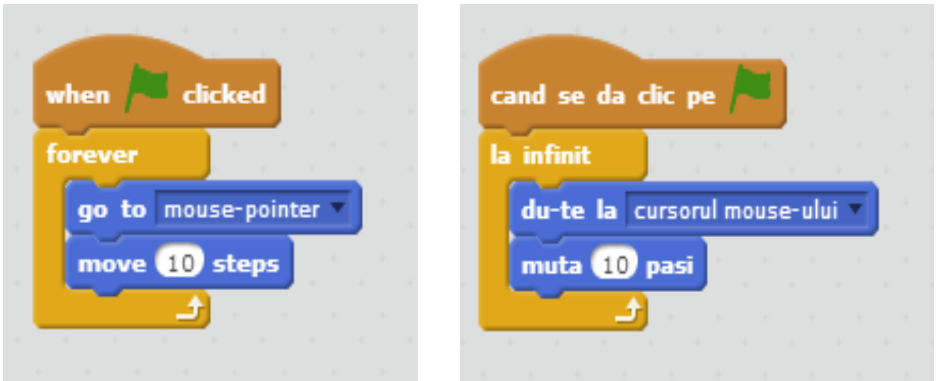


Fig.4.23 Personajul aleargă după cursorul mouse-ului.

Exemplu

Fig. 4.24 Personajul se îndreaptă spre personajul Creature1

Go to X: () Y: () – Se tastează câte un număr pentru x, respectiv y. Personajul se poziționează, pe scenă, la respectivele coordonate.

Exemplu

Fig. 4.25 Modificarea poziției personajului

Go to () – Poziționează personajul la coordonatele cursorului mouse-ului sau a altui personaj.

Exemple

Fig 4.26 Poziționarea personajului în dreptul cursorului la apăsarea "space"

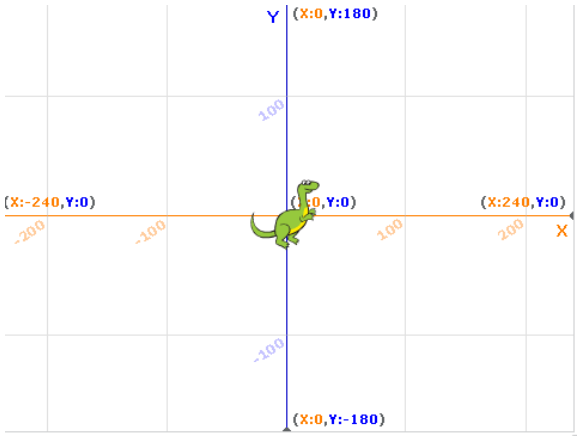


Fig. 4.27 Personajul în poziția inițială

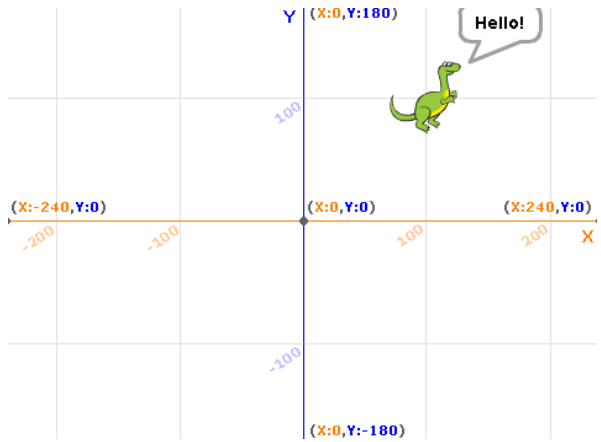


Fig. 4.28 Poziția personajului după rularea scriptului din figura Fig. 4.25

Fig. 4.29 Poziționarea personajului în dreptul personajului Creature1.

Glide () Secs to X: () Y: () – Personajul se mută în timpul specificat, la poziția precizată prin valorile date lui x și y.

Exemplu

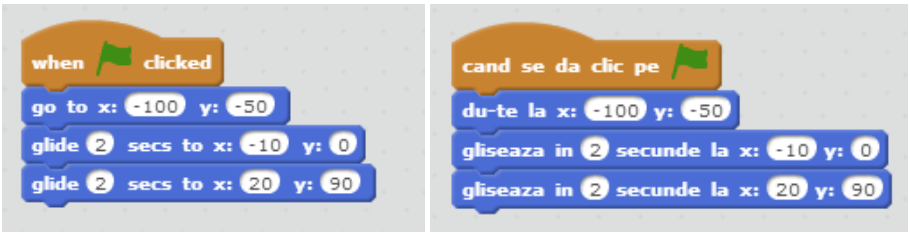


Fig. 4.30 Animație simplă

Change X by () – Modifică valoarea coordonatei x cu numărul specificat. Dacă numărul este pozitiv, personajul se mută la dreapta, iar dacă numărul este negativ personajul se va muta la stânga. (*Fig. 4.31*)

Change Y by () – Modifică valoarea coordonatei y cu numărul specificat. Dacă numărul este pozitiv, personajul se mută în sus, iar dacă numărul este negativ personajul se va mută în jos. (Fig. 4.32)

Exemplu

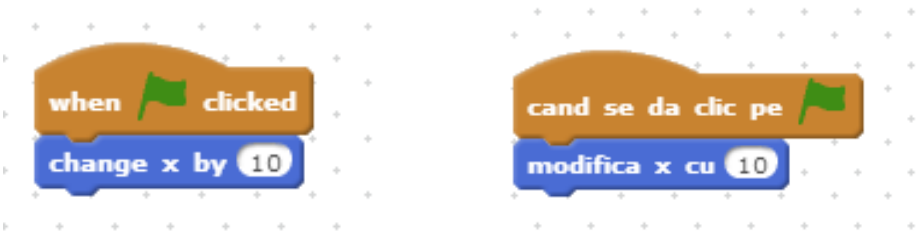


Fig. 4.31 Coordonata x se măreşte cu 10

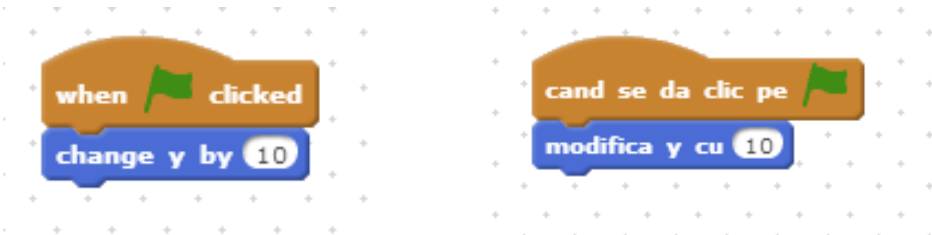


Fig. 4.32 Coordonata y se măreşte cu 10

Set X to () – Inițializează coordonata x a unui personaj.

Exemplu

Fig. 4.33 Personajul se mută în diferite puncte ale ecranului.

În urma rulării scriptului din Fig. 4.33 se vor produce rezultatele prezentate în Fig. 4.34, Fig. 4.35, Fig. 4.36 și în Fig. 4.37.

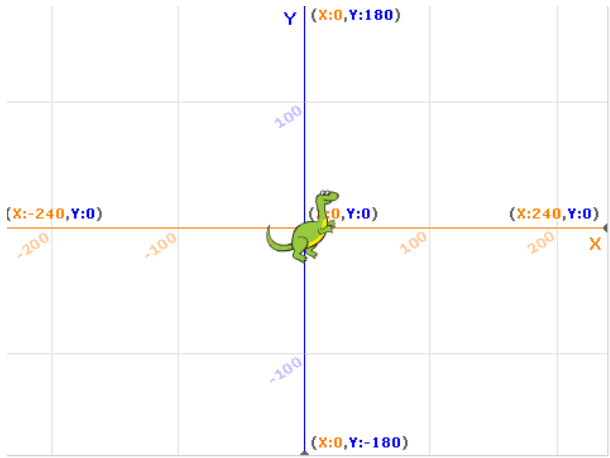


Fig. 4.34 Personajul în poziția inițială

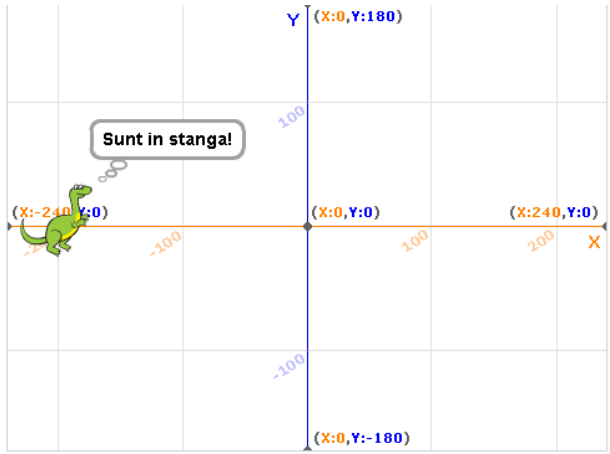


Fig. 4.35 Poziția personajului după modificarea coordonatei x la valoarea -200

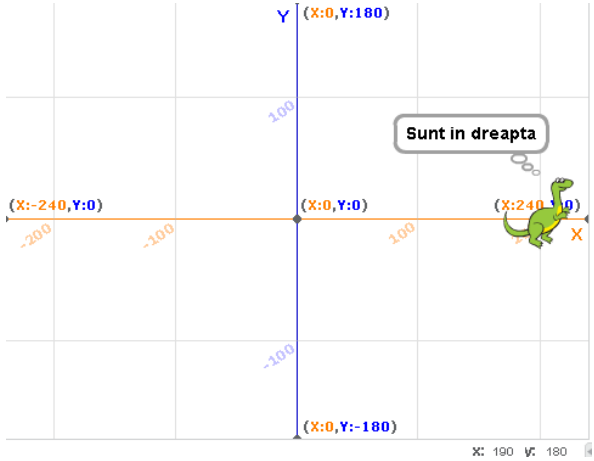


Fig. 4.36 Poziția personajului după modificarea coordonatei x la valoarea 200

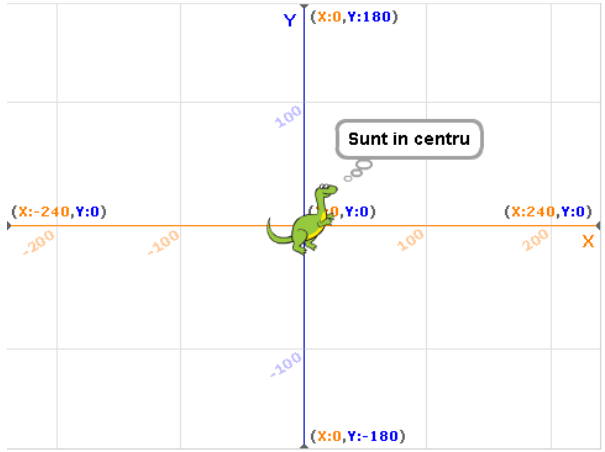


Fig. 4.37 Revenirea personajului la poziția inițială prin atribuirea valorii 0, coordonatei x

Set Y to () – Inițializează coordonata y a unui personaj.

Exemplu

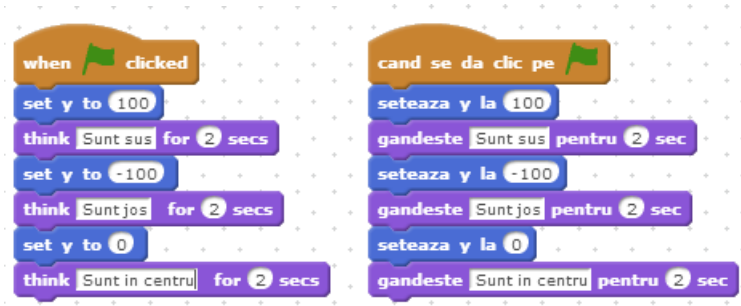


Fig. 4.38 Personajul se mută în diferite puncte ale ecranului.

În urma rulării scriptului din Fig. 4.38 se vor produce rezultatele prezentate în Fig. 4.39, Fig. 4.40, Fig. 4.41 și în Fig. 4.42.

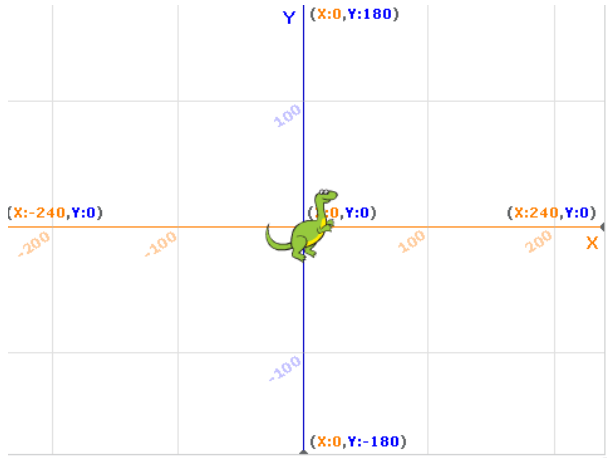


Fig. 4.39 Personajul în poziția inițială

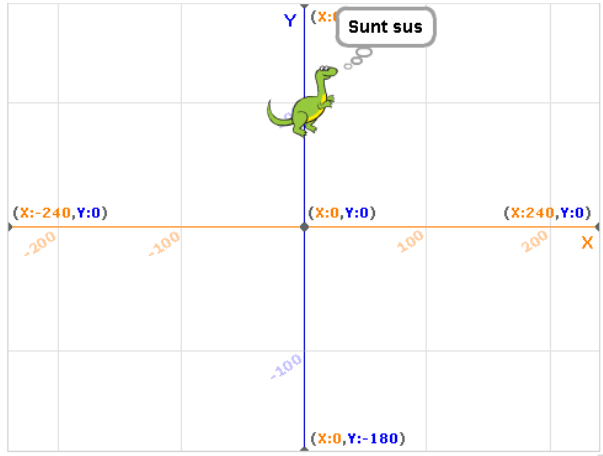


Fig. 4.40 Poziția personajului după modificarea coordonatei y la valoarea 100

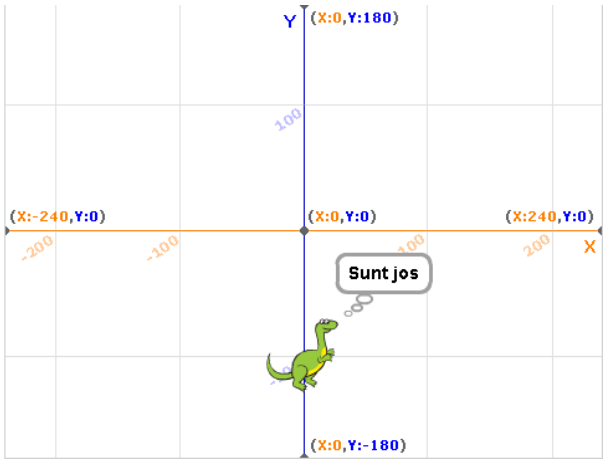


Fig. 4.41 Poziția personajului după modificarea coordonatei y la valoarea -100

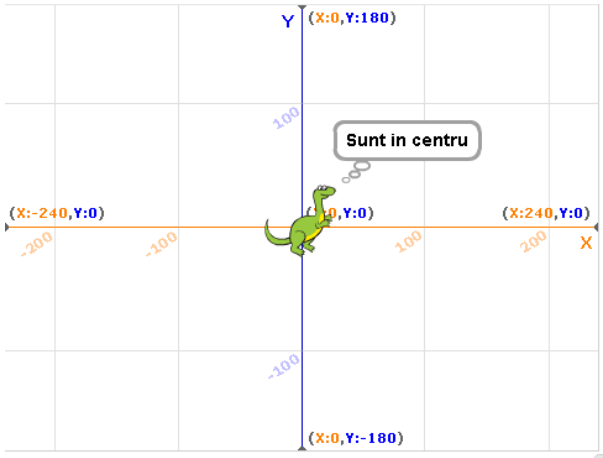


Fig. 4.42 Revenirea personajului la poziția inițială prin atribuirea valorii 0, coordonatei y

If on Edge, Bounce – Dacă personajul atinge marginea scenei, se întoarce. Comanda va fi exemplificată prin scriptul prezentat în Fig. 4.43.

X Position – Raportează coordonata x a personajului. (Se exemplifică în *Fig. 4.44*)

Y Position – Raportează coordonata y a personajului. (Se exemplifică în *Fig. 4.44*)

Exemple

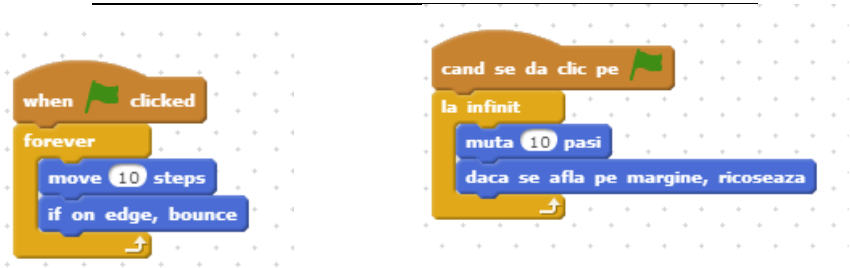


Fig. 4.43 Dacă personajul atinge marginea scenei, se intoarce

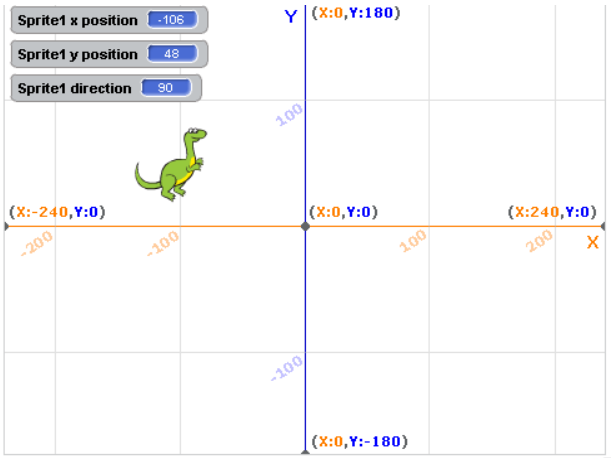


Fig. 4.44 Vizualizarea coordonatelor personajului prin selectarea opțiunilor X Position, Y Position

4.4 Blocurile LOOKS (Aspect)

Aceste blocuri sunt folosite pentru a controla aspectul unui personaj.

Say (mesaj) – Personajul transmite un mesaj într-un "balon de vorbire"

Fig. 4.45 Exemplu

Fig. 4.46 Balon de vorbire

Think (mesaj) – Se afișează mesajul specificat într-un "balon de gândire"

Fig. 4.47 Exemplu

Fig. 4.48 Balon de gândire

Think (mesaj) for (număr) Secs – Se afișează mesajul specificat într-un "balon de gândire" pentru un anumit număr de secunde. Numărul de secunde spune cât timp să fie vizibil "balonul de gândire". Script-ul așteaptă acel interval de timp înainte să continue.

Fig. 4.49 Exemplu

Switch to Costume ()/Switch to Backdrop () – blocul permite personajului/scenei să-și schimbe aspectul. Acest bloc, ca și următorul, este utilizat, în principal, în scripturi de animație. Pentru a îmbunătăți viteza de animație se folosește împreună cu blocul wait () secs.

Exemplu

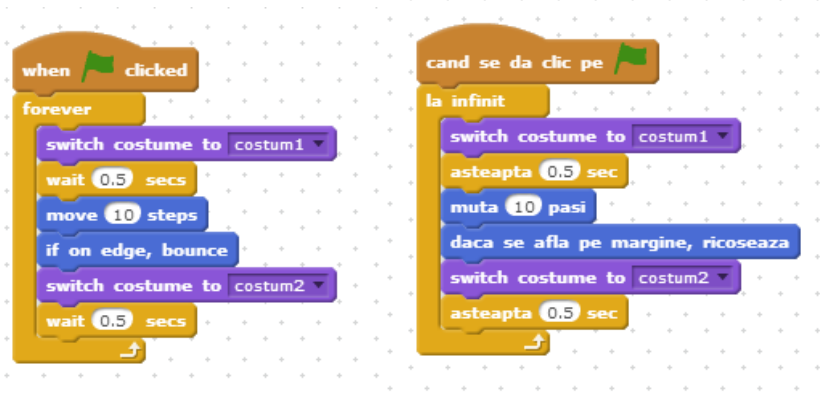


Fig. 4.50 Personajul își schimbă costumul la interval de 0.5 secunde.

Next Costume / Next Backdrop - Trece la următorul costum din lista de costume a personajului/scenei. În momentul în care se

ajunge la sfârșitul listei, se reia de la primul costum. Ordinea costumelor în listă poate fi modificată.

Fig. 4.51 Exemplu

Say (mesaj) for (număr) Secs – Personajul transmite un mesaj într-un "balon de vorbire" pentru un anumit număr de secunde. Numărul de secunde spune cât timp să fie vizibil "balonul de vorbire". Script-ul așteaptă acel interval de timp înainte să continue.

Utilizarea acestui bloc este exemplificată în Fig. 4.52.

Fig. 4.52 Exemplu

Set Size to (număr)% - Se setează dimensiunea personajului.

Show – Este utilizat atunci când se dorește ca un personaj să apară pe scenă.

Hide – Este utilizat atunci când se dorește ascunderea unui personaj.

Fig. 4.53 Exemplu blocuri show/hide

Change () Effect by () – Schimbă efectele grafice ale unui personaj. Există șapte efecte grafice predefinite: *color* (se schimbă culoarea personajului), *fisheye* (personajul pare a fi poziționat în apă), *whirl* (efect de vârtej), *pixelate, mosaic* (este creată iluzia mai multor personaje cu efect mozaic), *brightness* (crearea mai multor niveluri de luminozitate) și *ghost* (personaj transparent cu efect de fantomă). Aceste efecte pot fi selectate din meniul contextual.

Exemplu

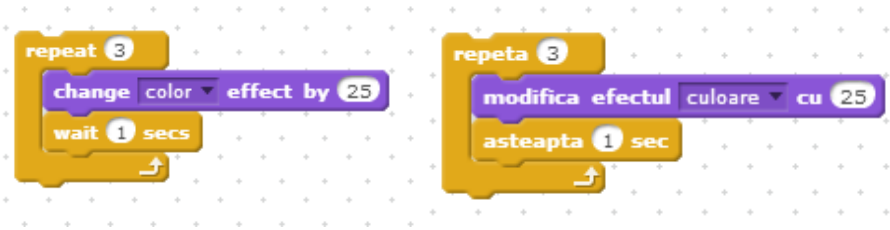


Fig. 4.54 Exemplu

În urma rulării scriptului prezentat în Fig. 4.54, personajul își va modifica culoarea în felul următor.

Clear Graphic Effects – Se elimină efectele grafice.

Exemplu

Fig. 4.55 Exemplu

Fig. 4.56 Personajul în forma inițială

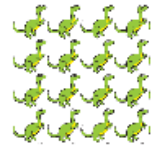


Fig. 4.57 Personajul după executarea scriptului prezentat în Fig. 4.55

Eliminarea efectelor grafice

Change Size by (număr) – Se modifică dimensiunea personajului.

Exemplu

Fig. 4.58 Modificarea dimensiunii personajului

Exemplu

Fig. 4.59 Exemplu

Fig. 4.60 Personajul in forma inițială

Fig. 4.61 Personajul după executarea scriptului prezentat în Fig. 4.59

Set () Effect to () – Setează efectele grafice ale unui personaj.

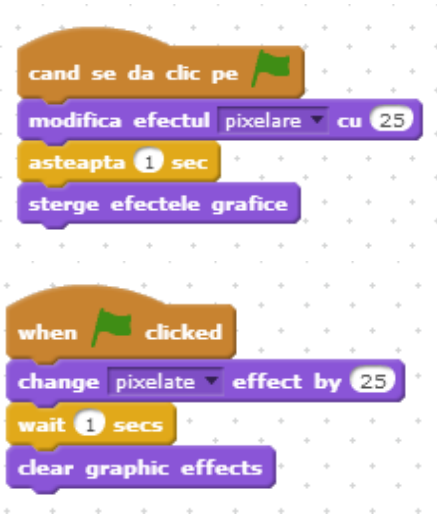


Fig. 4.62 Exemplu

Fig. 4.63 Personajul în forma inițială

Fig. 4.64 Personajul după executarea scriptului prezentat în Fig. 4.62

Efectele vor fi selectate din meniul contextual al blocului.

Exemplu

Fig. 4.65 Realizarea unor efecte speciale, utile în realizarea animațiilor

Fig.4.66 Personajul in forma inițială

Fig. 4.67 Personajul după executarea scriptului prezentat în Fig. 4.65

Exemplu

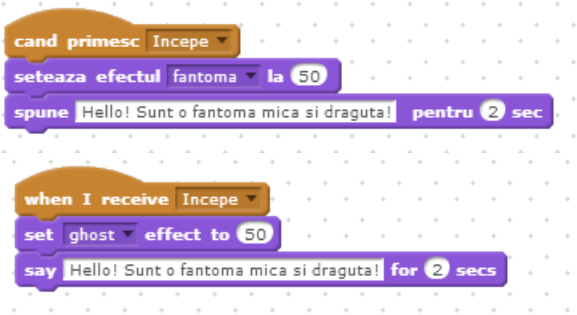


Fig. 4.68 Realizarea unor efecte speciale

Fig. 4.69 Personajul in forma inițială

Fig. 4.70 Personajul după executarea scriptului prezentat în Fig. 4.68

Go to Front – Personajul este mutat în fața tuturor celorlalte personaje.

Exemplu

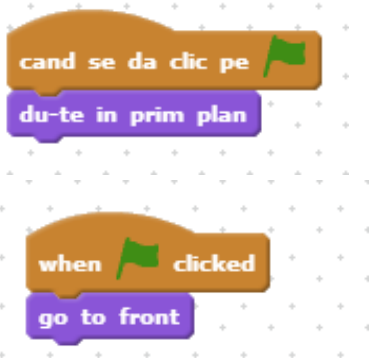


Fig. 4.71 Exemplu

Fig. 4.72 Personajul in forma inițială

Fig. 4.73 Personajul după executarea scriptului prezentat în Fig. 4.71

Go Back (număr) Layers – Personajul este ascuns în spatele altor personaje (al căror număr este specificat).

Exemplu

Fig. 4.74 Exemplu

Fig. 4.75 Personajul in forma inițială

Fig. 4.76 Personajul după executarea scriptului prezentat în

4.5 Blocurile SOUND (Sunet)

Aceste blocuri sunt utilizate pentru a controla funcțiile de sunet și MIDI.

Play Sound () – Acest bloc este utilizat pentru redarea unui sunet. Sunetul este redat până la sfârșitul acestuia chiar dacă programul trece la executarea următorului bloc.

Play Sound () Until Done – Acest bloc este utilizat pentru redarea unui sunet. Înainte de a se trece la următorul bloc, se așteaptă până când se termină melodia.

Stop All Sounds – Oprește toate sunetele.

Play Drum () for () Beats – Redă un sunet de tobă.

Play Note () for () Beats – Redă o notă muzicală.

Nota muzicală este codificată printr-un număr natural din intervalul [0,127]. Cu cât numărul ales este mai mare, cu atât nota muzicală este mai înaltă.

Set Instrument to () – Setează instrumentul muzical folosit de personaj pentru a reda notele muzicale.

Instrumentul poate fi selectat dintr-un meniu contextual sau prin tastarea unui număr natural din intervalul [1,21].

Un personaj poate utiliza numai un singur instrument la un moment dat. Pentru a reda mai multe instrumente simultan, este necesar a avea mai multe personaje.

Exemplu

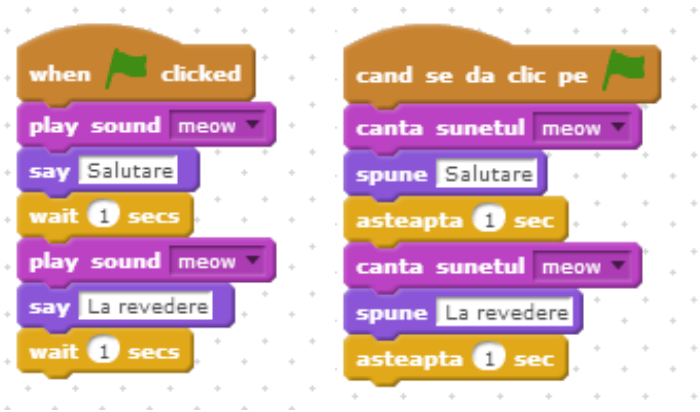


Fig. 4.77 Redarea unui sunet în timpul unei acțiuni.

Exemplu

Fig. 4.78 Realizarea fundalului sonor

Exemplu

Fig. 4.79 Înainte de a rula sunetul "pop", toate sunetele vor fi oprite la primirea mesajului "new song".

Exemplu

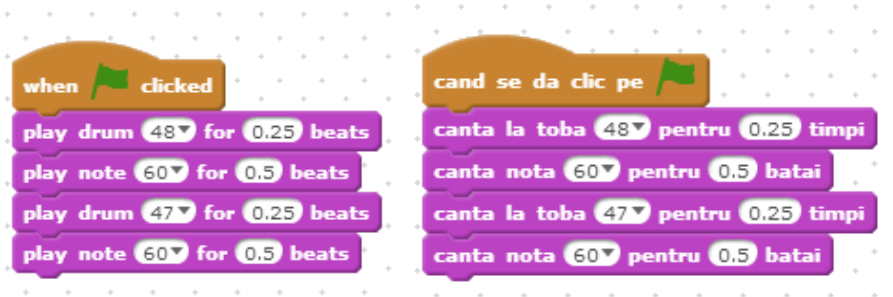


Fig. 4.80 Compunerea unui cântec

Exemplu

Fig. 4.81 Redarea unor note muzicale la apăsarea tastei "space".

4.6 Blocurile PEN (Penița)

Aceste blocuri sunt utile atunci când dorim să realizăm anumite desene în cadrul unui proiect. Utilizarea blocurilor Pen nu influențează fundalul sau personajele. Urmele de stilou și ștampile nu fac parte din fundal, de aceea, realizarea sau curățarea lor nu afectează fundalul.

Cu ajutorul acestor blocuri se pot realiza scripturi cu rezultate spectaculoase.

Pen Down – Personajul va realiza un traseu (va lăsa o urmă de penița) ori de câte ori se mișcă (până se folosește blocul Pen Up). Culoarea, lățimea și umbra traseului pot fi schimbate cu alte blocuri independente.

Pen Up – Anulează efectul blocului Pen Down.

Clear – Se șterg toate desenele

Stamp – Se imprimă imaginea personajului.

Set Pen Color to (culoare) – Setează culoarea peniței. Culoarea va fi selectată din meniul contextual.

Change Pen Color by (număr) – Culorile sunt codificate prin numere naturale cuprinse în intervalul [0,100]. Utilizarea acestui bloc duce la modificarea culorii curente prin incrementarea

(mărirea)/ decrementarea (micșorarea) codului cu numărul specificat.

Set Pen Color to (number) – Setează culoarea peniței. Culoarea va fi codificată printr-un număr natural cuprins în intervalul [1,200].

Change Pen Size by (număr) – Modifică lățimea urmei lăsate de penița. Lațimea urmei este codificată prin numere naturale cuprinse în intervalul [0,255].

Set Pen Size to (număr) – Setează lățimea urmei lăsate de penița.

Change Pen Shade by (număr) – Intensitatea unei culori este codificată prin numere naturale cuprinse în intervalul [0,100]. Utilizarea acestui bloc duce la modificarea intensității culorii curente prin incrementarea (mărirea)/ decrementarea (micșorarea) codului cu numărul specificat.

Exemplu

Fig. 4.82 Exemplu

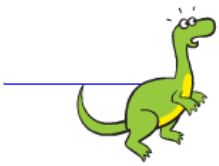


Fig. 4.83 Rezultat în urma rulării scriptului prezentat în Fig. 4.82

Exemplu

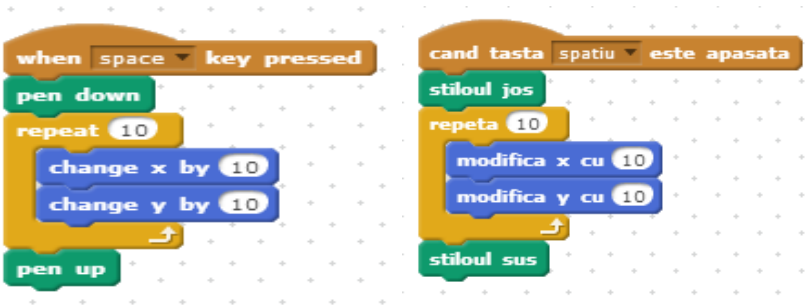


Fig. 4.84 Exemplu

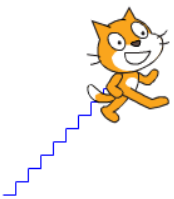


Fig. 4.85 Rezultat în urma rulării scriptului prezentat în Fig. 4.84

Exemplu

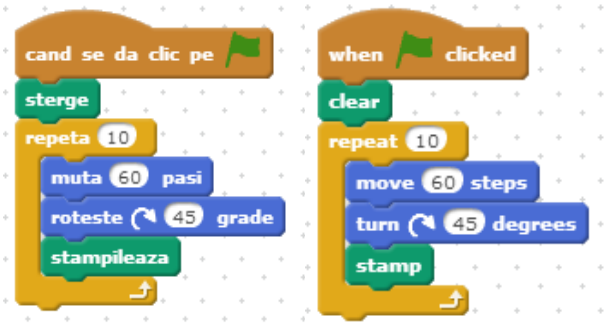


Fig. 4.86 Exemplu

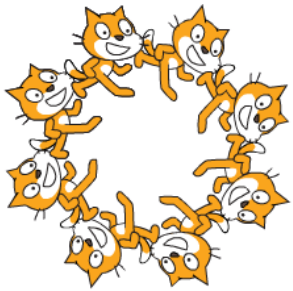


Fig. 4.87 Rezultat în urma rulării scriptului prezentat în Fig. 4.86

Exemplu

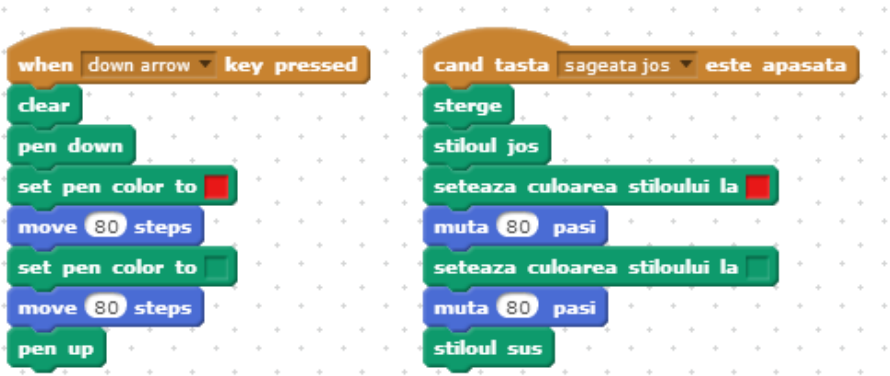


Fig. 4.88 Exemplu

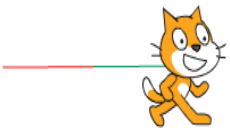


Fig. 4.89 Rezultat în urma rulării scriptului prezentat în Fig. 4.88

Exemplu

Fig. 4.90 Exemplu

Fig. 4.91 Rezultat în urma rulării scriptului prezentat în Fig. 4.90

Exemplu

Fig. 4.92 Exemplu

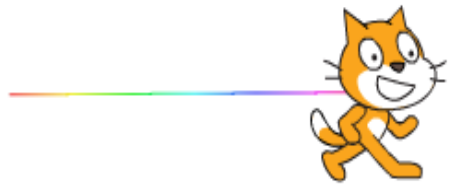


Fig. 4.93 Rezultat în urma rulării scriptului prezentat în Fig. 4.92

Exemplu

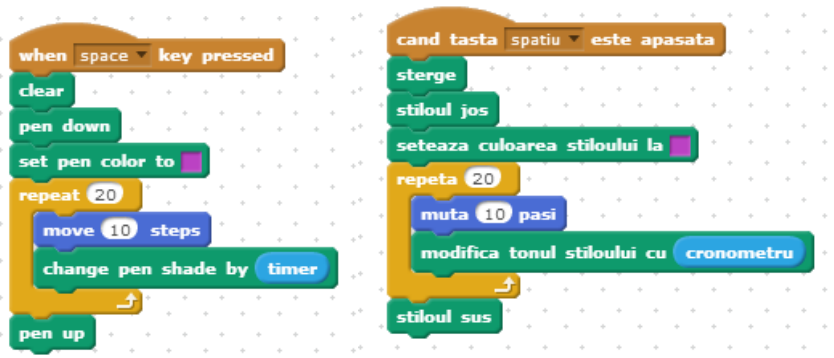


Fig. 4.94 Exemplu

Fig. 4.95 Rezultat în urma rulării scriptului prezentat în Fig. 4.94

Exemplu

Fig. 4.96 Exemplu

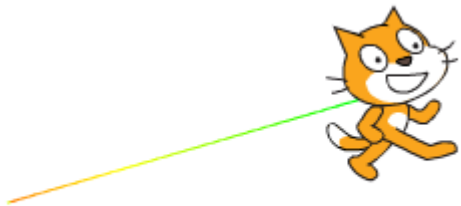


Fig. 4.97 Rezultat în urma rulării scriptului prezentat în Fig. 4.96

Exemplu

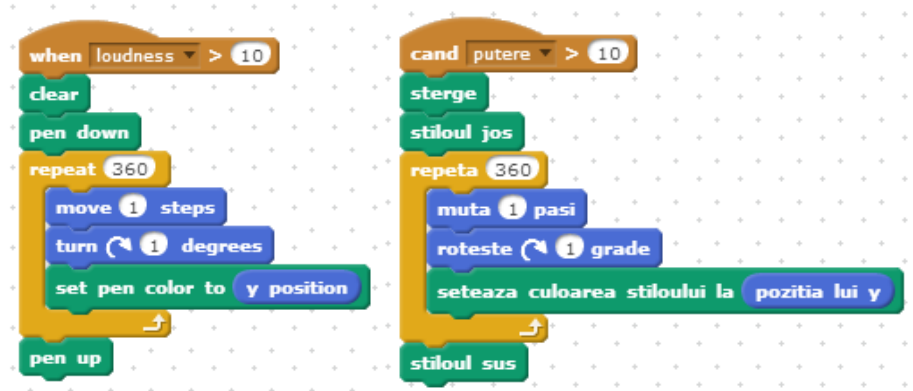


Fig. 4.98 Exemplu

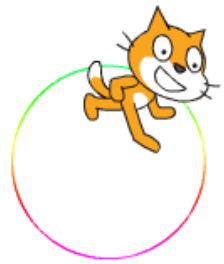


Fig. 4.99 Rezultat în urma rulării scriptului prezentat în Fig. 4.98

Exemplu

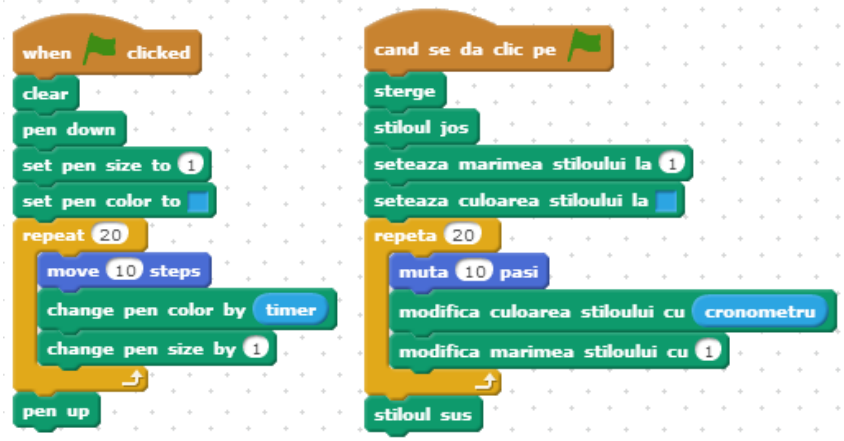


Fig. 4.100 Exemplu

Fig. 4.101 Rezultat în urma rulării scriptului prezentat în Fig. 4.100

Exemplu

Fig. 4.102 Exemplu

Fig. 4.103 Rezultat în urma rulării scriptului prezentat în Fig. 4.102

Exemplu

Fig. 4.104 Exemplu

Fig. 4.105 Rezultat în urma rulării scriptului prezentat în Fig. 4.104

Exemplu

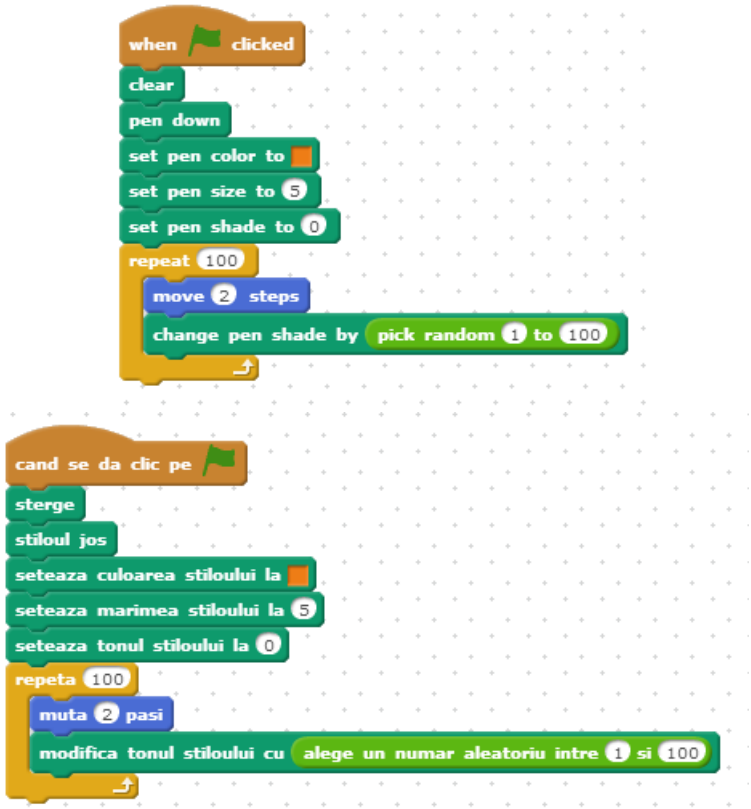


Fig. 4.106 Exemplu

Fig. 4.107 Rezultat în urma rulării scriptului prezentat în Fig. 4.106

Exemplu

Fig. 4.108 Exemplu

Fig. 4.109 Rezultat în urma rulării scriptului prezentat în Fig. 4.108

4.7 Blocurile DATA (Date)

Aceste blocuri sunt utilizate atunci când dorim să introducem variabile în programe.

Permite crearea unei noi variabile. Când se creează o variabilă, vor apărea blocurile specifice ce permit

prelucrarea variabilei. Variabila poate fi recunoscută de un singur personaj (variabilă locală) sau poate fi recunoscută de toate personajele din proiect (variabilă globală).

Permite ștergerea unei variabile și a tuturor blocurilor asociate cu acestea.

Arată valoarea unei variabile.

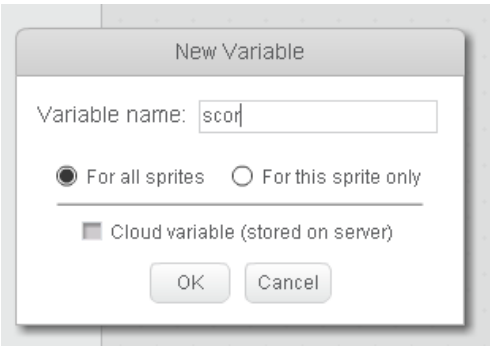


Fig. 4.110 Declararea variabilei scor, recunoscuta de toate personajele proiectului

Set (nume_variabila) to (număr) – Inițializează valoarea variabilei cu valoarea specificată.

Change (nume_variabila) by (număr) – Modifică valoarea variabilei cu valoarea specificată.

Show Variable (nume_variabila) – Valoarea variabilei va fi afișată pe scenă.

Hide Variable (nume_variabila) – Valoarea variabilei nu mai este afișată.

Exemplu

Fig. 4.111 Exemplu

Fig. 4.112 Rezultat în urma rulării scriptului prezentat în Fig. 4.111

Exemplu

Fig. 4.113 Exemplu

Fig. 4.114 Rezultat în urma rulării scriptului prezentat în Fig. 4.113

Exemplu

Fig. 4.115 Exemplu

Fig. 4.116 Rezultat în urma rulării scriptului prezentat în fig. Fig. 4.115

4.8 Blocurile SENSING (Senzori)

Ask () and Wait – Acest bloc este foarte util atunci când utilizatorul trebuie să comunice cu proiectul. Blocul permite afișarea pe ecran a unei întrebări și memorează intrarea de la tastatură (răspunsul) în variabila:

După primirea unui răspuns, programul va aștepta până când este apăsată tasta Enter sau până se dă clic pe caseta de validare. În felul acesta îi este permis utilizatorului să dea o comandă.

Exemplu

Fig. 4.117 Exemplu

Fig. 4.118 Rezultat în urma rulării scriptului din Fig. 4.117

Exemplu

Fig. 4.119 Primirea informațiilor de la utilizator

Fig. 4.120 Rezultat în urma rulării scriptului din Fig. 4.119

Exemplu

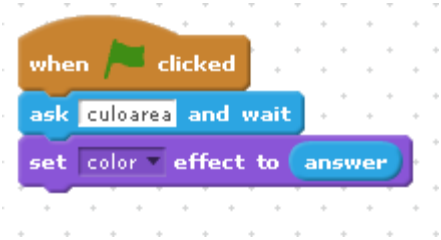


Fig. 4.121 Stabilirea preferințelor - coordonate, culori - de către utilizator

Fig. 4.122 Rezultat în urma rulării scriptului prezentat în Fig. 4.121

Touching ()? - Acest bloc returnează TRUE (ADEVĂRAT) dacă personajul atinge marginea scenei, cursorul mouse-ului sau un alt personaj

specificat. Acest bloc se comportă diferit atunci când personajul este este ascuns. Astfel dacă personajul ascuns atinge un alt personaj, blocul are valoarea FALSE și are valoarea TRUE (ADEVĂRAT) dacă simte pointerul mouse-ului și marginile.

Exemple

Fig. 4.123 Mutarea unui personaj până când atinge marginea ecranului

Fig. 4.124 Personajul aleargă către un alt personaj până îl prinde

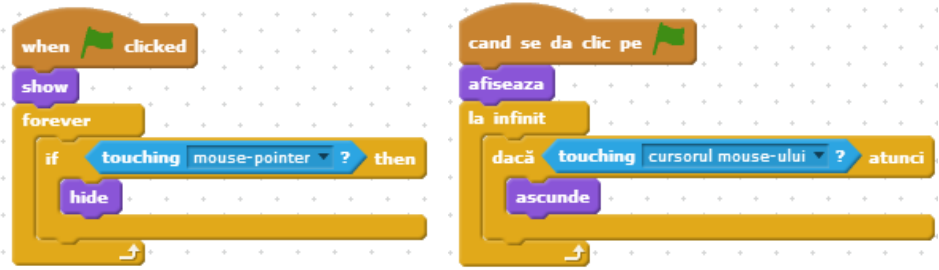
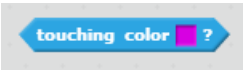


Fig. 4.125 Personajul dispare atunci când este atins de mouse-ul pointerului.

Touching Color ()? – Are valoarea TRUE (ADEVĂRAT) dacă personajul atinge culoarea precizată. 

Exemple

Fig. 4.126 Personajul reacționează într-un mod particular atunci când se atinge de culoarea verde

Fig. 4.127 Personajul reacționează într-un mod particular atunci când se atinge culoarea albastru

Fig. 4.128 Personajul se mișcă până atinge un "perete" de o anumită culoare.

Fig. 4.129 Personajul dispare atunci când atinge un "perete" de o anumită culoare.

Color (culoare1) is Touching (culoare2)? – Rezultatul este TRUE (ADEVĂRAT) dacă culoare1 atinge culoare2

Exemple

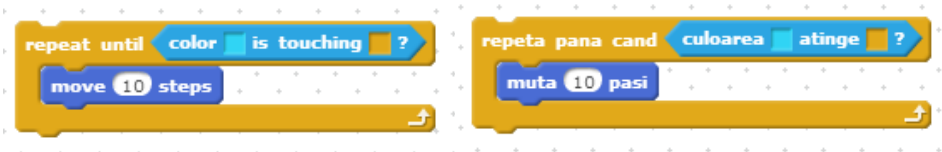


Fig. 4.130 Personajul se mișcă până când o culoare a costumului său atinge o altă culoare

Fig. 4.131 Atunci când o culoare a costumului personajului atinge o altă culoare se întâmplă un anume eveniment.

Key (tasta) Pressed? Blocul verifică dacă s-a apăsat o anumită tastă. Rezultatul este TRUE (ADEVĂRAT) dacă tasta specificată este apăsată. Tastele ce pot fi verificate vor fi cele alfanumerice (litere a-z, cifre 0-9), taste cu săgeți (down arrow, up arrow, left arrow, right arrow) și tasta space.

Exemplu

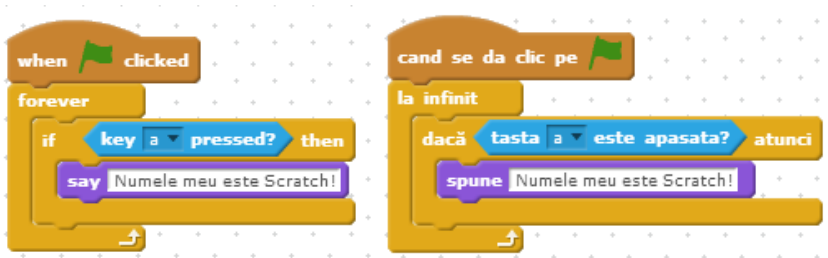


Fig. 4.132 Atunci când tasta "a" este apăsată, personajul își spune numele.

Fig. 4.133 Rezultatul scriptului prezentat în Fig. 4.132

Mouse Down? Rezultatul este TRUE (ADEVĂRAT) dacă butonul mouse-ului este apăsat. 

Exemplu

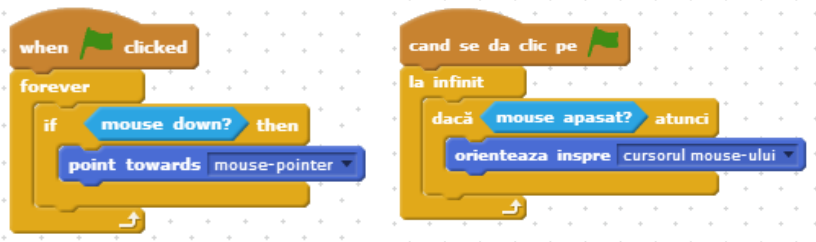


Fig. 4.134 Personajul se poziționează la coordonatele pointerului mouse-ului, atunci când acesta este apăsat.

Distance to () Determină distanța (măsurată în pixeli) dintre personajul curent și un alt personaj sau dintre personajul curent și pointerul mouse-ului.

Exemplu

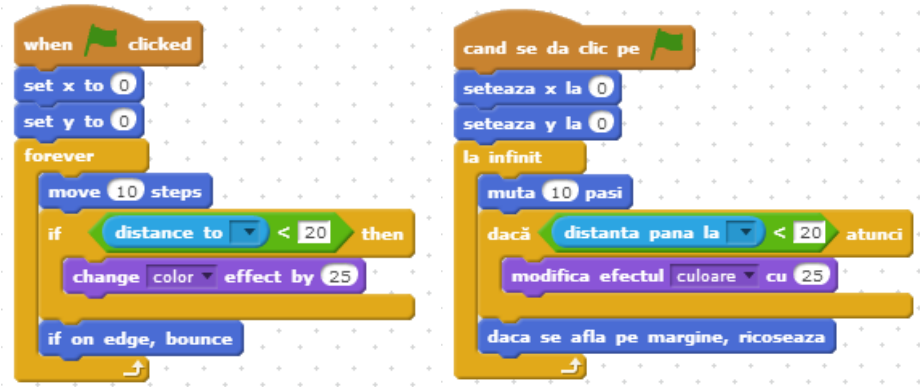


Fig. 4.135 Costumul personajului își modifică culoarea atunci când acesta se află la o distanța mai mică de 20 de pixeli față de pointerul mouse-ului.

Mouse X Determină poziția x a cursorului mouse-ului. 

Mouse Y Determină poziția y a cursorului mouse-ului. 

Ultimele două blocuri pot fi utilizate ca operanzi în expresii.

4.9 Blocurile OPERATORS (Operatori)

4.9.1 Operatori logici

(expresie1) < (expresie2) 

Are valoarea TRUE (ADEVĂRAT) dacă valoarea expresiei1 este mai mică decât valoarea expresiei 2 și valoarea FALSE în caz contrar.

Exemplu

Fig. 4.136 Dacă distanța dintre două personaje este mai mică de 10 pixeli, atunci personajul curent va face 10 pași.

(expresie1) = (expresie2) 

Are valoarea TRUE (ADEVĂRAT) dacă valoarea expresiei1 este egală cu valoarea expresiei2 și valoarea FALSE în caz contrar.

Exemplu

Fig. 4.137 Verificarea valorii unei variabile.

(expresie1) > (expresie2) 

Are valoarea TRUE (ADEVĂRAT) dacă valoarea expresiei1 este mai mare decât valoarea expresiei2 și valoarea FALSE în caz contrar.

Exemplu

Fig. 4.138 Modificarea scorului unui joc

not (expresie) 

Are valoarea TRUE (ADEVĂRAT) dacă expresia este falsă.

Exemplu

Fig. 4.139 Verificarea valorii unei variabile

(expresie1) and (expresie2) 

Are valoarea TRUE (ADEVĂRAT) dacă ambele expresii sunt adevărate.

Exemplu

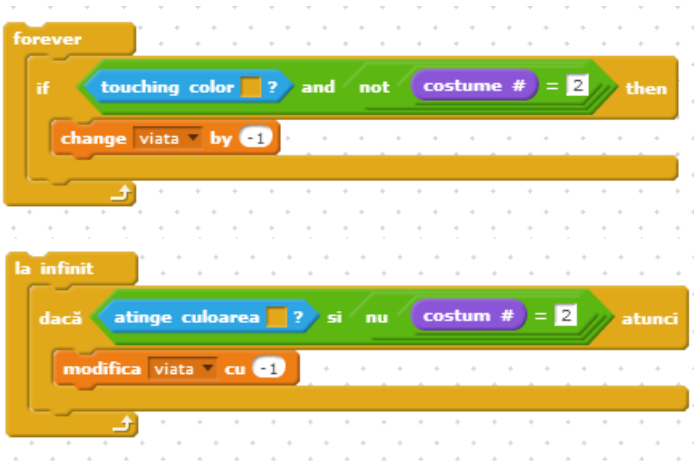
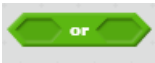


Fig. 4.140 Personajul pierde o viață dacă atinge zidul și nu are costumul adecvat.

(expresie1) or (expresie2) 

Are valoarea TRUE (ADEVĂRAT) dacă cel puțin una dintre expresii este adevărată.

Exemplu

Fig. 4.141 Simularea conversației dintre un personaj și utilizator.

4.9.2 **Operatori aritmetici**

() + () - Adună două numere

() - () - Scade două numere

() * () - Înmulțește două numere

() / () - Împarte două numere

() Mod () - Restul împărțirii a două numere

Round () - Cel mai apropiat întreg de un număr

Exemple

Fig. 4.142 Exemplu

Fig. 4.143 Rezultatul inițial al scriptului din Fig. 4.142

Fig. 4.144 Rezultatul obținut după 2 secunde

Fig. 4.145 Exemplu

Fig. 4.146 Rezultatul scriptului din Fig. 4.145

(functie matematica) of (număr) – Sunt aplicate diferite funcții matematice.

Fig. 4.147 Funcții matematice disponibile

Exemplu

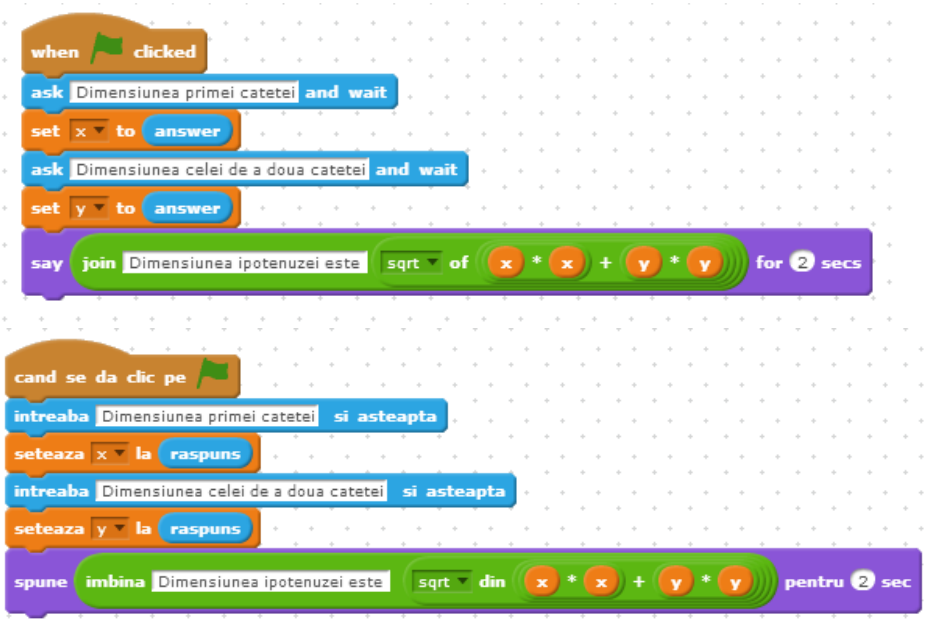


Fig. 4.148 Teorema lui Pitagora: Utilizatorul introduce dimensiunile catetelor, iar personajul va spune dimensiunea ipotenuzei

4.9.3 **Alți operatori**

Pick Random (număr1) to (număr2) .

Este ales, la întâmplare, un număr întreg din intervalul determinat de număr1 și număr2;

Join (sir1)(sir2)

Concatenează (lipește) șir2 la sfârșitul lui șir1

Exemplu

Fig. 4.149 Exemplu

Fig. 4.150 Rezultatul scriptului prezentat în Fig. 4.149

Exemplu

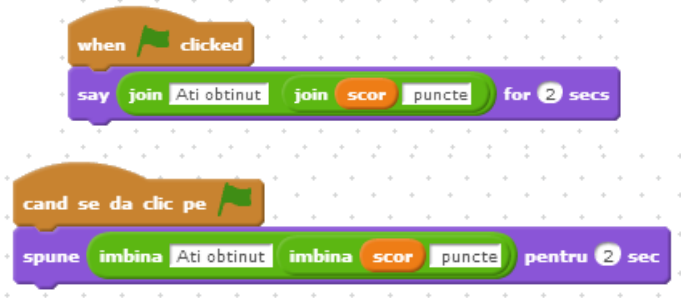


Fig. 4.151 Concatenarea şirurilor de caractere cu valoarea unei variabile

Fig. 4.152 Rezultatul scriptului prezentat în Fig. 4.151

Exemplu

Fig. 4.153 Exemplu

Fig. 4.154 Rezultatul scriptului prezentat în Fig. 4.153

Length of (șir)

Determină numărul caracterelor din șirul dat.

Letter (număr) of (șir)

Determină litera din șirul dat, de pe poziția specificată de număr.

Exemplu

Fig. 4.155 Exemplu

Fig. 4.156 Rezultatul scriptului prezentat în Fig. 4.155

Exemplu

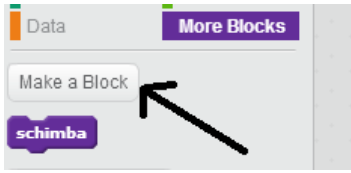
Rezultate – la intervale de o secundă

4.10 Blocuri personalizate (More blocks)

Blocuri personalizate - proceduri

Există posibilitatea ca o secvență de program să fie scrisă o singură dată și utilizată de mai multe ori.

Acest lucru este posibil dacă se va crea un bloc nou denumit procedură. O procedură este creată făcând clic pe opțiunea “Make a Block”, din cadru “More Blocks”.



Se va deschide o fereastră de dialog, în care vom putea denumi noul bloc. Făcând clic pe OK se va putea completa, în zona de script, corpul procedurii.

În cazul în care procedura este apelată, se vor rula blocurile din cadrul acesteia.

Exemplu

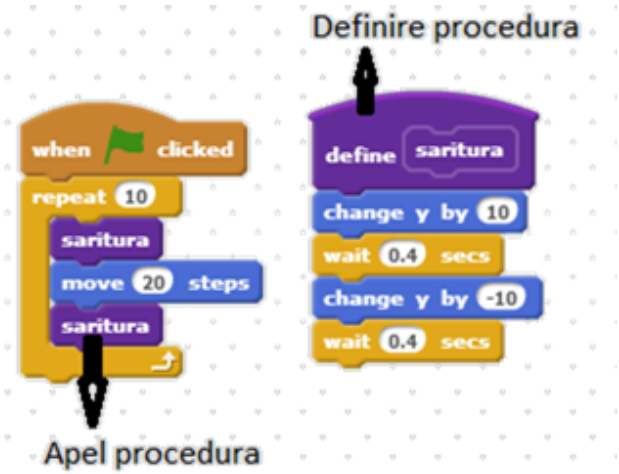


Fig. 4.157 Realizarea unui efect de animație

Exemplu

Fig. 4.158 Realizarea efectului de "coborâre scară". Blocul personalizat realizează desenarea unei singure trepte. Blocul va fi folosit în mod repetat

Capitolul 5. **Programe**

Acum, pentru că le ştim pe toate, să trecem la treabă!

5.1 Cum realizăm primul program în Scratch?

Utilizatorii pot realiza programe în Scratch, aplicând tehnica **drag-and-drop** asupra blocurilor de acţiune din paleta de blocuri şi ataşarea lor la alte blocuri sub forma unui puzzle logic.

Structurile de blocuri multiple se numesc scripturi. Această metodă de programare (realizarea codului prin utilizarea blocurilor) este cunoscută sub denumirea de **"drag-and-drop programming"** (programare prin selectare, tragere şi plasare).

Ne propunem sa creăm un program care să afișeze imaginile din Fig. 5.1.

Fig. 5.1 Pisica dansatoare

Primii pași pentru a realiza acest prim program sunt prezentați în Fig. 5.2 și în Fig. 5.3.

Fig. 5.2 Se aduce un bloc de mișcare (MOVE) în zona de program.

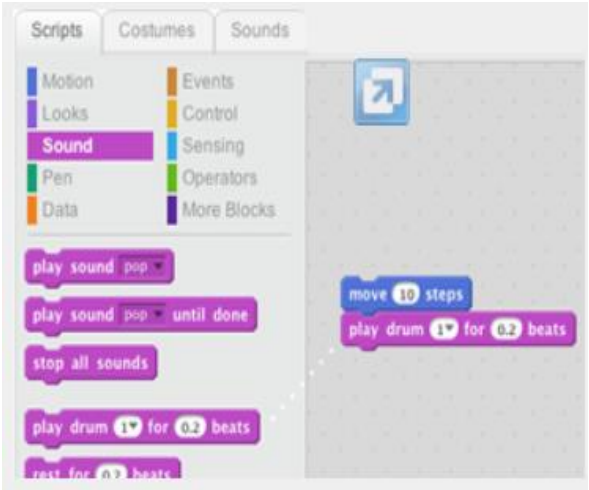


Fig. 5.3 Trageți un bloc PLAY DRUM (de la sunet) și fixați-l pe blocul MOVE

Se adaugă un alt bloc MOVE. Faceți clic în interiorul blocului și tastați un semn minus.

Se adaugă un alt bloc PLAY DRUM, apoi alegeți un tambur din meniul pull-down.

Trageți un bloc REPEAT și fixați-l pe partea de sus a stivei.

Programul va rula în momentul în care se apasă steguleț verde.

În timpul dansului, personajul își poate schimba culoarea dacă programul va fi modificat ca în Fig. 5.6.

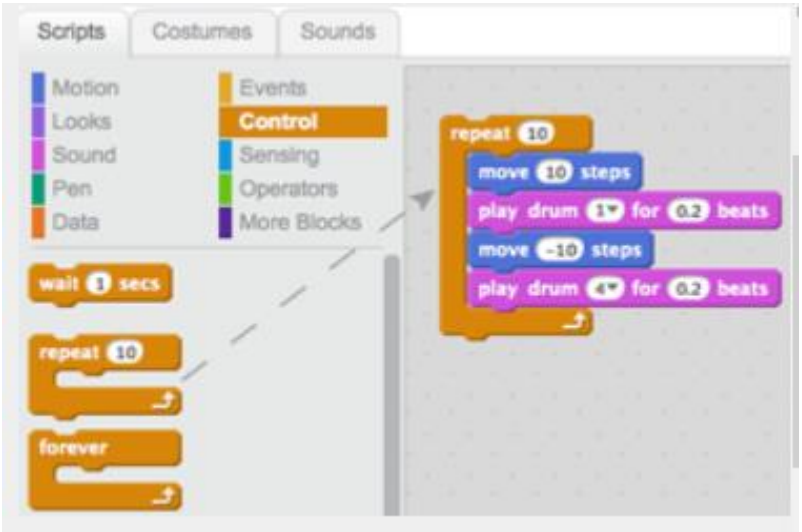


Fig. 5.4 Blocul REPEAT trebuie să conțină toate cele patru blocuri.

Fig. 5.5 Programul în forma finală

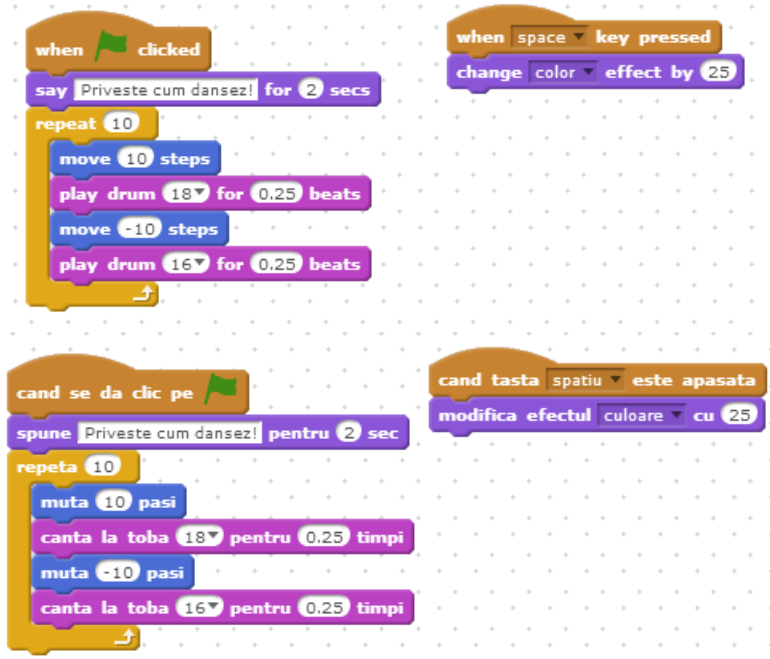


Fig. 5.6 La apăsarea tastei „space” pisica își modifică culoarea

5.2 Crearea efectelor

5.2.1 Schimbarea culorii

Se va folosi blocul:

Din meniu se poate selecta atât culoarea, cât și un alt efect.

Efectul ales poate fi încercat cu numere întregi din intervalul [-100,100].

Program

Pentru un prim exemplu, se alege un personaj multicolor.

Fig. 5.7 Personajul își schimbă culoarea atunci când este apăsată tasta "space".

Fig. 5.8 Personajul în formă inițială

Fig. 5.9 Rezultatul executării scriptului din Fig. 5.7

Program

Fig. 5.10 Script ce modifică de patru ori culoarea personajului, la interval de 1 secundă

Efectul acestui program va fi următorul:

Inițial | După 1 secundă | După 2 secunde | După 3 secunde

5.2.2 Efecte interactive

Program

Fig. 5.11 Personajului îi sunt aplicate efecte, în funcție de poziția cursorului.

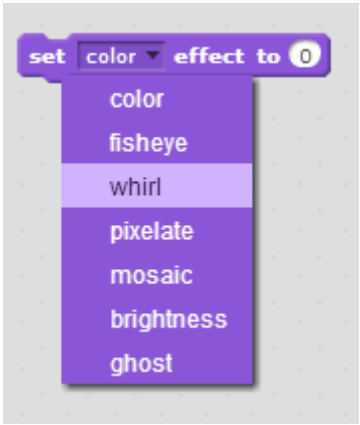


Fig. 5.12 Se pot selecta diverse efecte din meniul pull-down, asociate cu numere întregi din intervalul [-100, 100].

5.2.3 Efecte create cu ajutorul microfonului

Utilizand secvența prezentată în Fig. 5.13, personajul se va mișca în sus în momentul în care se produce zgomot. Cu cât zgomotul este mai puternic, cu atât personajul va sări mai mult.

Fig. 5.13 Producerea unui zgomot poate genera o scurtă animație

5.3 Animații

5.3.1 Schimbarea costumelor

În timpul execuției programelor, personajele își pot schimba costumele astfel încât să reflecte diferite stări sufletești, să redea mișcarea sau să dea impresia că vorbesc.

5.3.2 Trecerea de la o stare la alta

Se creează sau se alege un personaj cu două costume ca în Fig. 5.14 .

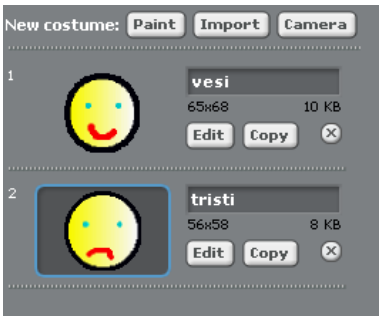


Fig. 5.14 Alegerea personajului

Scriptul este prezentat în Fig. 5.15.

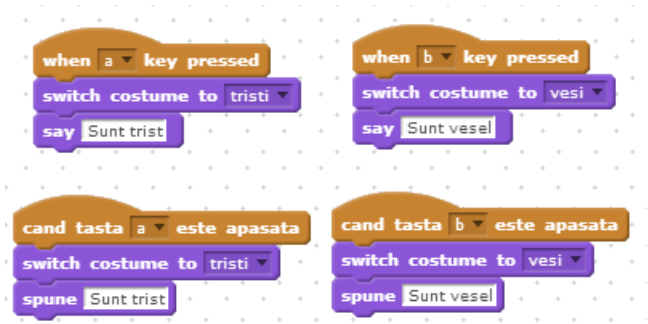


Fig. 5.15 Personajul își schimbă costumul la apăsarea tastelor "a" și "b"

5.3.3 Realizarea efectului de vorbire

Va fi ales un personaj cu două costume – cu gura deschisă și cu gura închisă ca în Fig. 5.16. Scriptul este prezentat în Fig. 5.17.

Fig. 5.16. Alegerea personajului

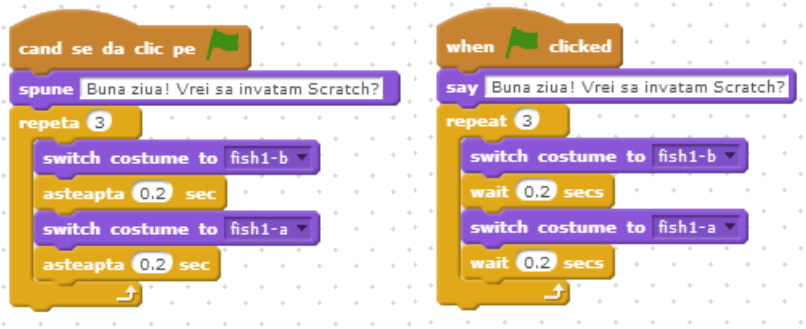


Fig. 5.17 Realizarea efectului de vorbire

5.3.4 **Realizarea efectului de deplasare - mers**

Va fi ales un personaj cu costume, în care acesta este redat în mai multe ipostaze ca în Fig. 5.18. Scriptul este prezentat în Fig. 5.19.

Fig. 5.18 Alegerea personajulu

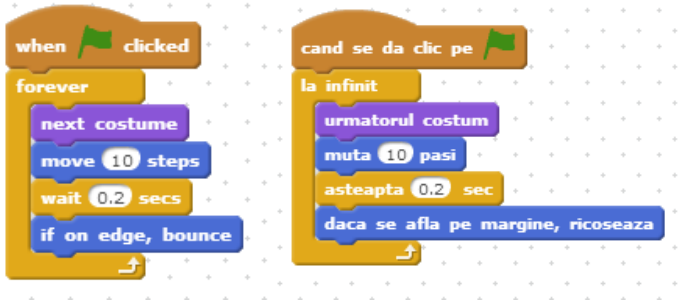


Fig. 5.19 Realizarea efectului de deplasare

Pentru a fixa modul în care personajul se întoarce în momentul în care ajunge la marginea scenei, se utilizează opțiunile:

5.3.5 **Controlul personajelor cu ajutorul săgeților**

Scriptul prezentat în Fig. 5.21 permite controlul mișcării unui personaj cu ajutorul săgeților.

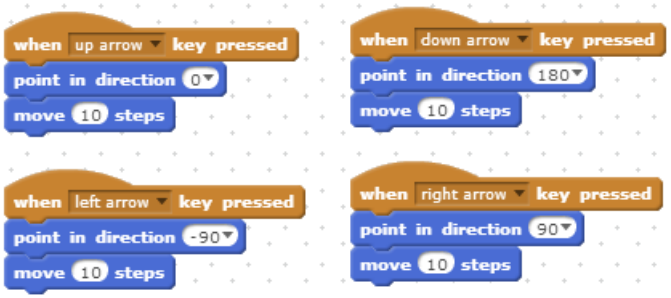
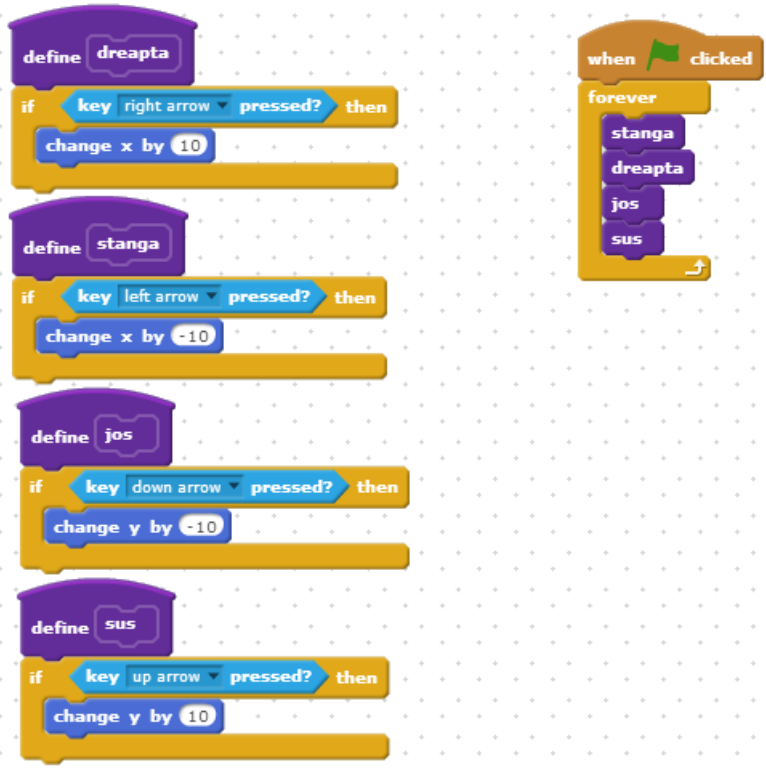


Fig. 5.20 Utilizarea săgeților pentru a controla un personaj

Controlul mișcării stânga/dreapta, respectiv sus/jos se poate realiza prin modificarea coordonatei x, respectiv y, ca în scriptul prezentat în Fig. 5.22.

Vor fi definite 4 blocuri noi. Fiecare dintre acestea va controla personajul prin intermediul uneia dintre tastele săgeți.

Fig. 5.21 Utilizarea săgeților pentru a controla un personaj



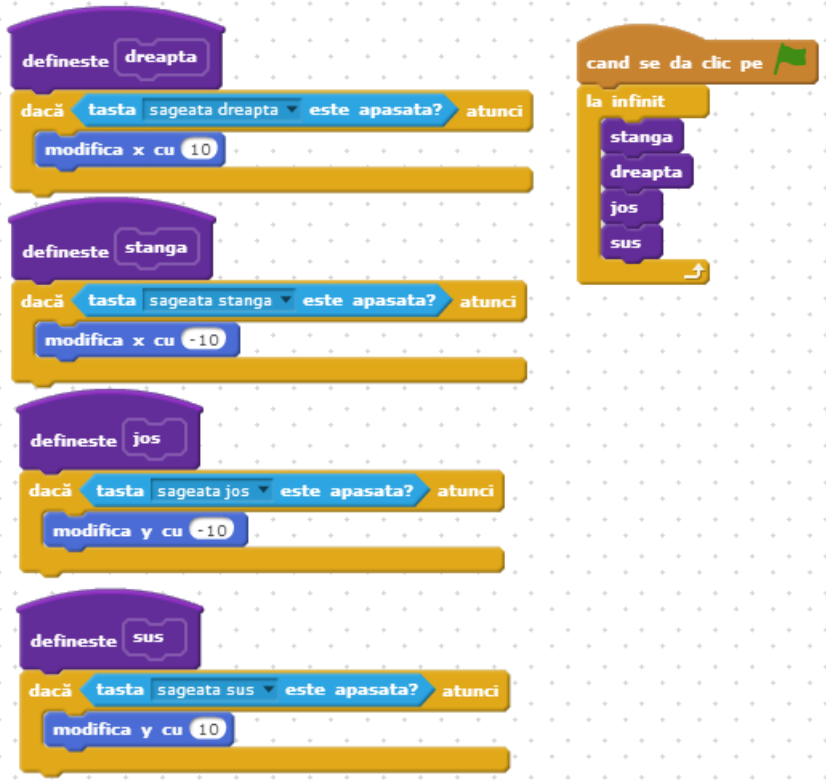


Fig. 5.22 Modificarea coordonatelor

Exemplu

În scriptul prezentat în Fig. 5.222 se dorește realizarea unui joc: ieșirea dintr-un labirint. Mingea este controlată cu ajutorul săgeților. Ea se va putea deplasa stânga-dreapta, jos-sus.

Personajul are la început un număr de vieți. În momentul în care mingea se lovește de un perete, se va pierde o viață.

Jocul poate fi dezvoltat astfel încât dacă mingea atinge dreptunghiul jucătorul va trece la un alt nivel.

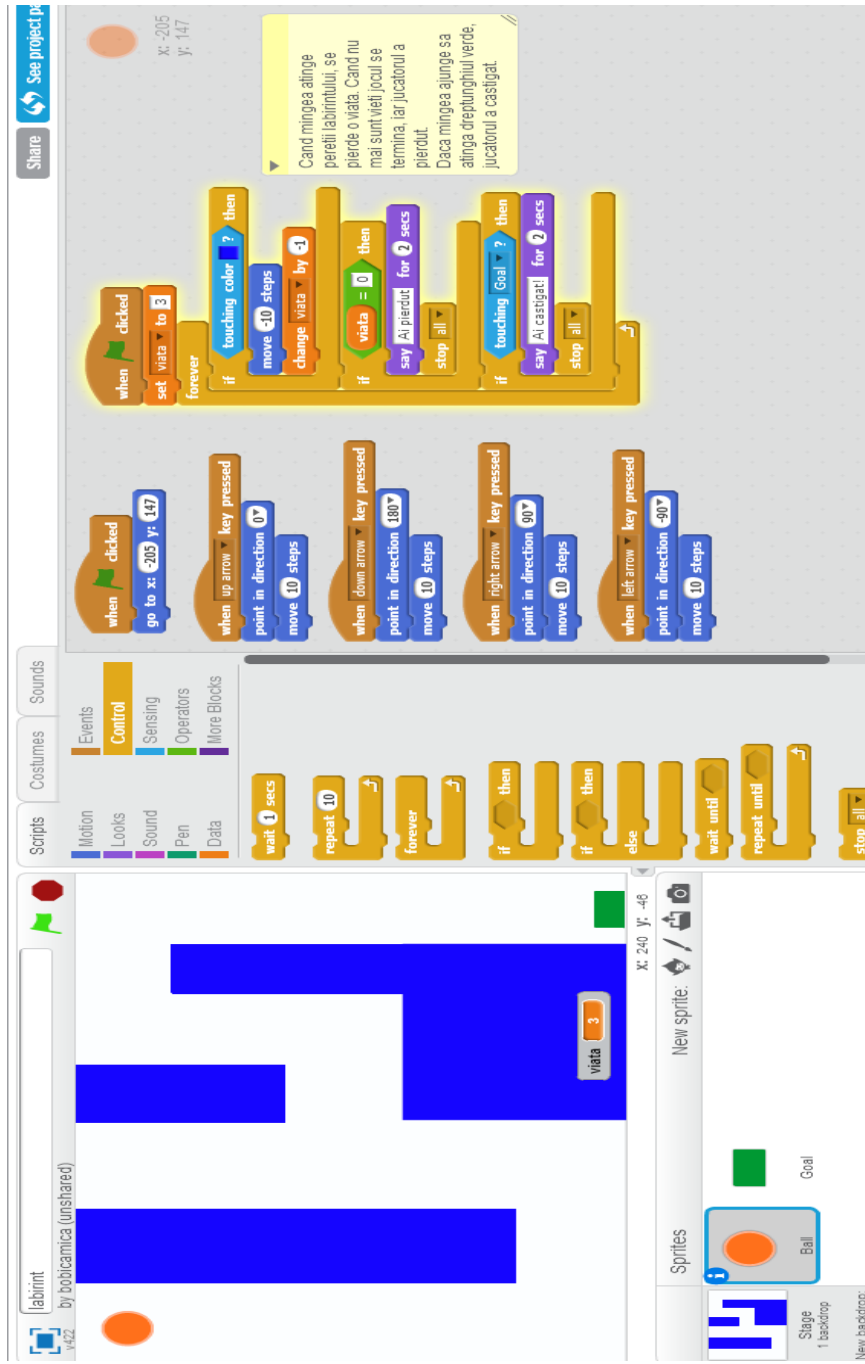


Fig. 5.23 Ieşirea dintr-un labirint.

5.3.6 Simularea unei sărituri

Program

Fig. 5.24 Simularea unei sărituri

Program

Fig. 5.25 Simularea unuei sărituri cu aterizare pe o platformă

Program

Scriptul prezentat în Fig. 5.26 realizează un efect realist pentru sărituri. Este folosită o variabilă pentru a controla viteza pe verticală a personajului.

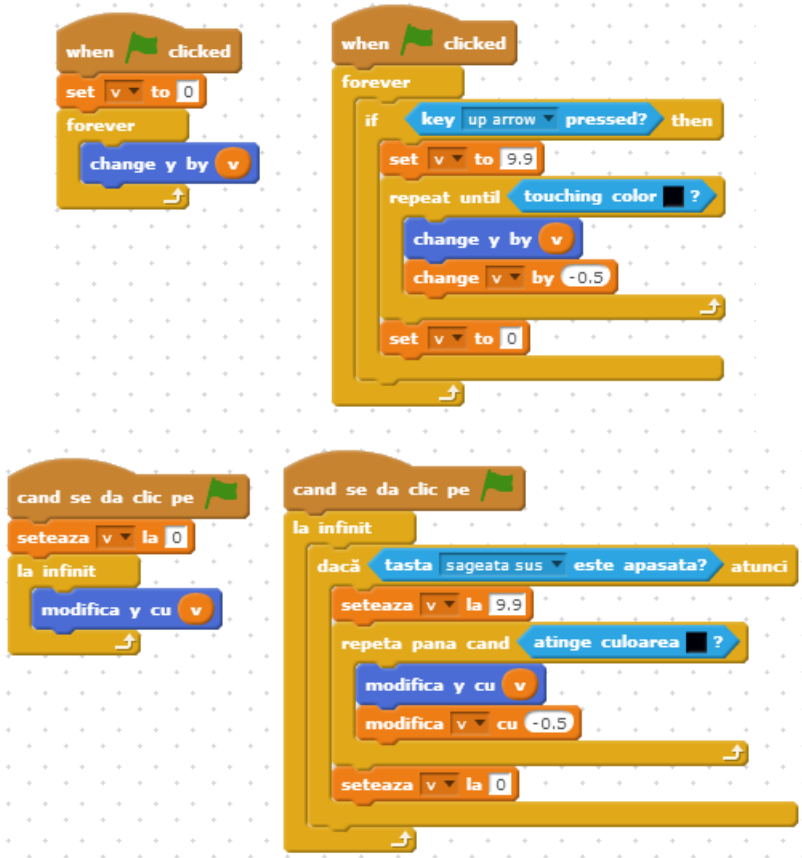


Fig. 5.26 Simularea gravitației

5.4 Interacțiunea personajelor

Primul pas îl reprezintă crearea personajelor redate în Fig. 5.27

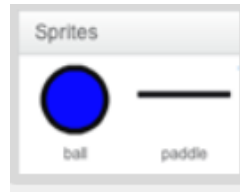


Fig. 5.27 Personaje

Corespunzător personajului „ball” se scrie scriptul prezentat în Fig. 5.28.

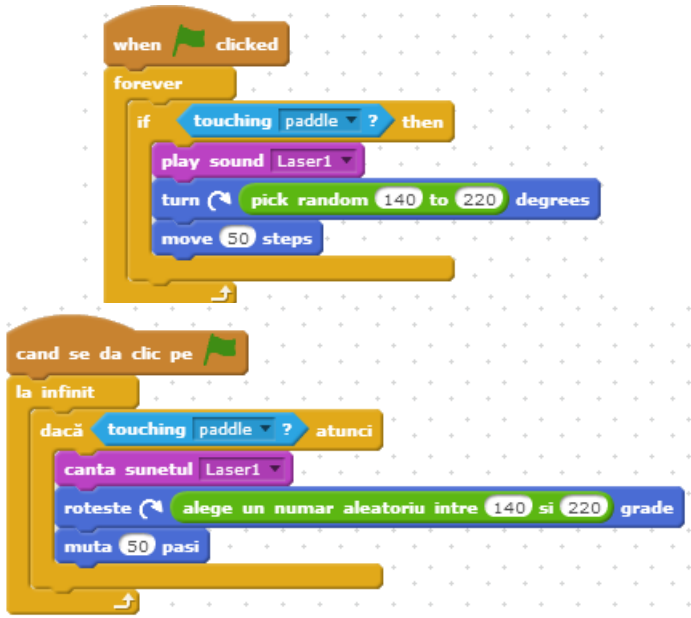


Fig. 5.28 Realizarea interacțiunii dintre personaje

5.5 Scorul unui joc

Putem crea jocuri în care este necesară păstrarea scorului.

Pentru a realiza un astfel de program, se va declara o variabilă pentru a reține scorul.

Se selecteaza blocul Variables și apoi se apasă opțiunea Make a variable.

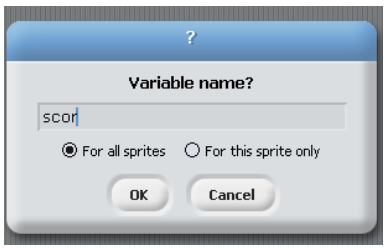


Fig. 5.29 Se tastează numele variabilei și apoi se apasă ok.

Sunt necesare două personaje (Scratch si Dog Puppy). Urmează adăugarea de blocuri în zona de script, corespunzătoare personajului Scratch, pentru a obține programul din Fig. 5.30.

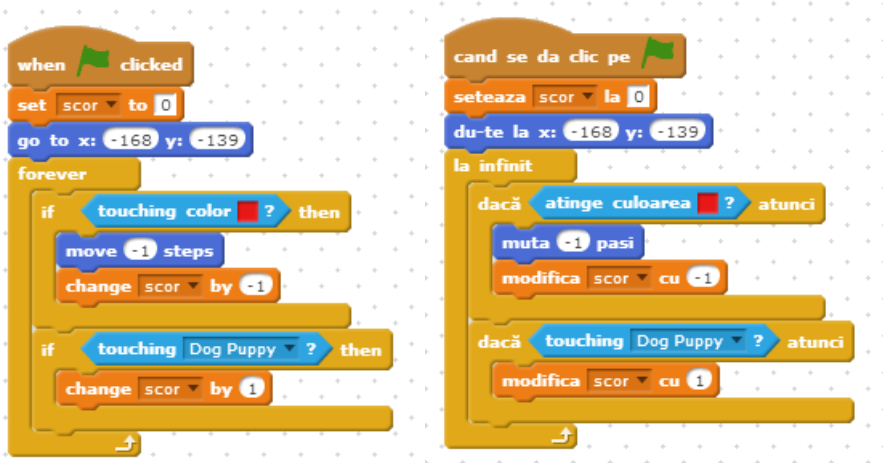


Fig. 5.30 Modificarea scorului unui joc

5.6 Simularea unei conversații între două personaje

Blocul „broadcast” transmite mesaje între personaje. În momentul în care unul dintre personaje primește un mesaj, va ști că poate „vorbi”. Pentru a realiza conversația prezentată în Fig. 5.31, se adăugă blocurile prezentate în Fig. 5.32 și în Fig. 5.33, separat pentru fiecare personaj în parte.

Fig. 5.31 Simularea conversației dintre două personaje

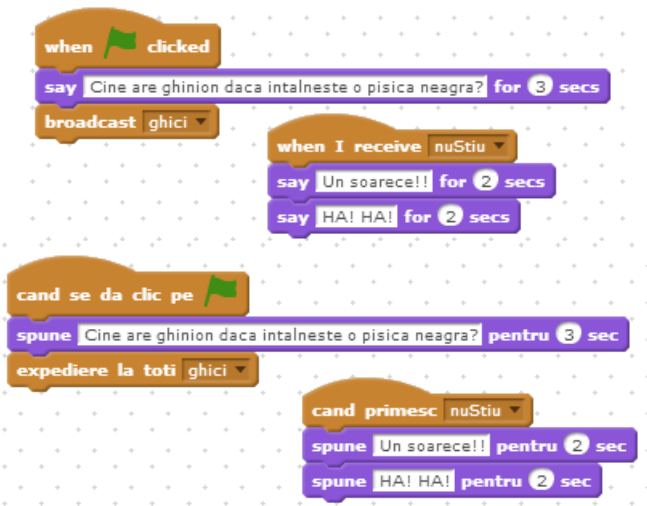


Fig. 5.32 Script corepunzător Personajului1 - pisica

Fig. 5.33 Script corespunzător Personajului2 – cățelul

5.7 Ieșirea din scenă a unui personaj

Program

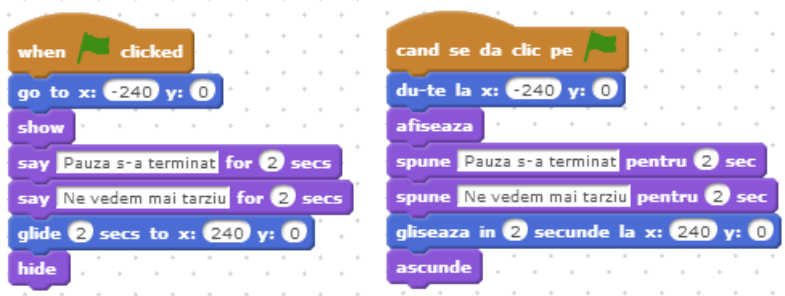


Fig. 5.34 Ieșirea din scenă a unui personaj

Program

În anumite situații este nevoie ca personajul sa evolueze în decoruri diferite. Într-un astfel de caz se pot utiliza următoarele secvențe scrise atât în zona de script a personajului, cât și în zona de script a scenei.

Fig. 5.35 Blocuri din zona de script corespunzătoare personajului

Fig. 5.36 Blocuri din zona de script corespunzătoare scenei

5.8 Interacțiunea cu utilizatorul

Un personaj poate interacționa cu utilizatorul prin intermediul blocului: 

. Răspunsul introdus de la tastatură este memorat în variabila predefinită: 

Program

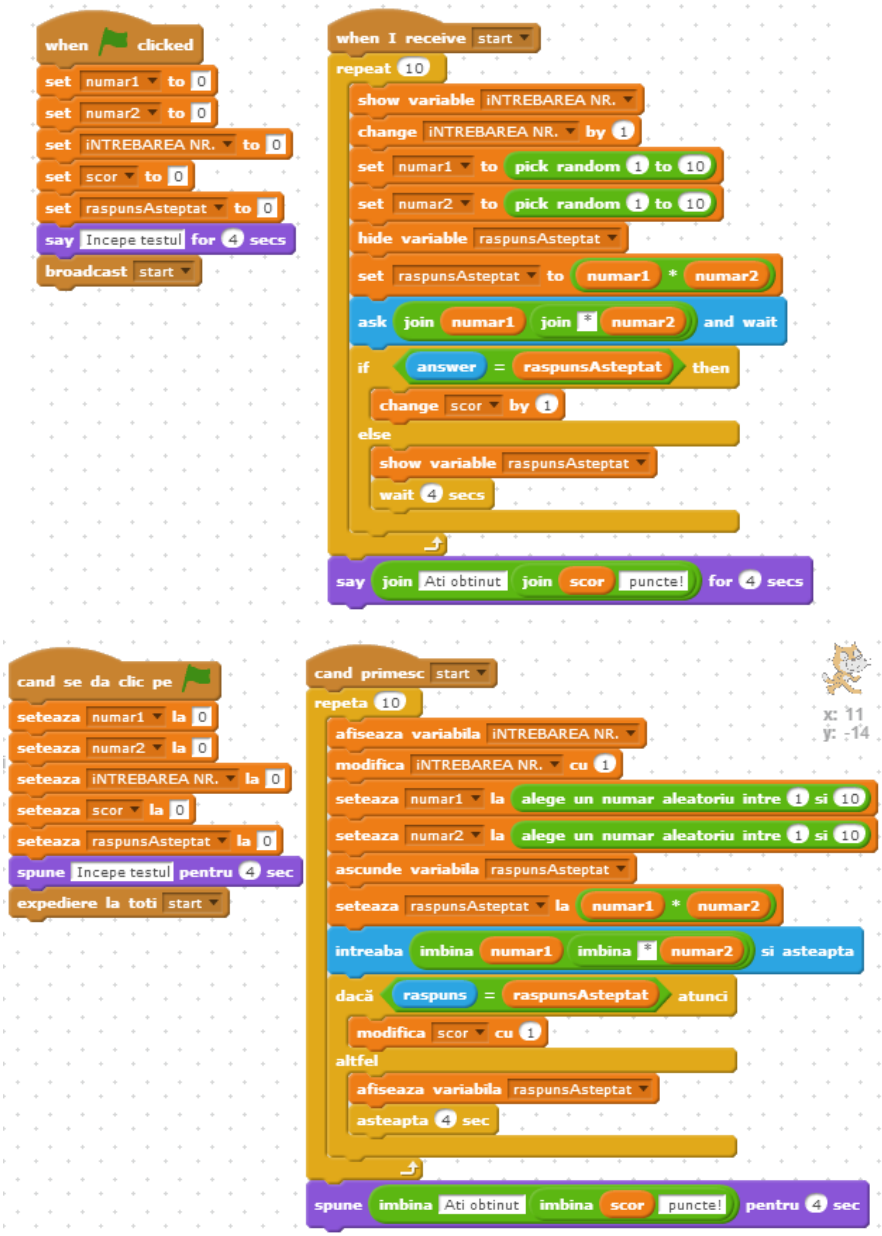


Fig. 5.37 Test pentru a verifica modul în care utilizatorul stăpânește tabla înmulțirii

5.9 Interacțiunea cu mouse-ul

Următoarele două scripturi pot fi utilizate ca punct de plecare pentru realizarea unor butoane de meniu.

Programe

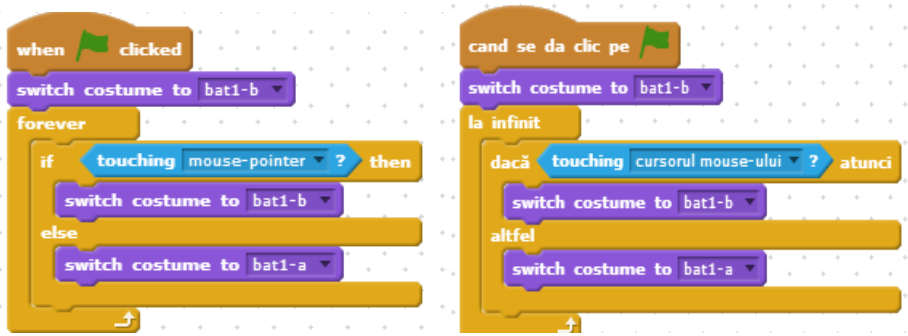


Fig. 5.38 Schimbarea costumului unui personaj în momentul în care acesta este atins de cursorul mouse-ului.

Fig. 5.39 Schimbarea luminozității personajului în momentul în care butonul drept al mouse-ului este apăsat.

5.10 Realizarea unor desene

Program

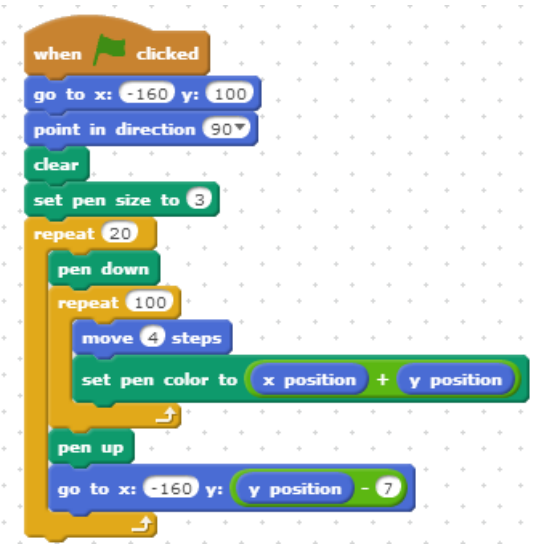


Fig. 5.40 Personajul va desena, schimbând culoarea pensulei în funcție de poziția sa

Program

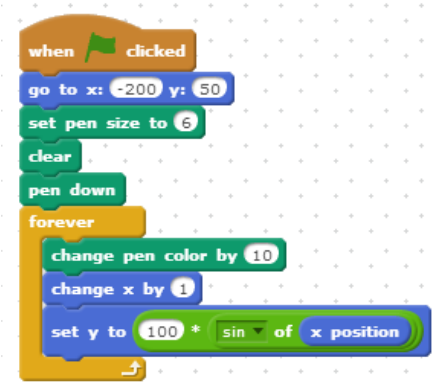


Fig. 5.41 Script realizat cu ajutorul funcțiilor trigonometrice

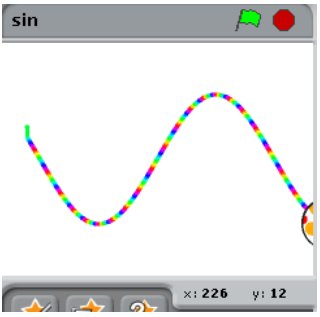


Fig. 5.42 Rezultatul scriptului din Fig. 5.41

Program

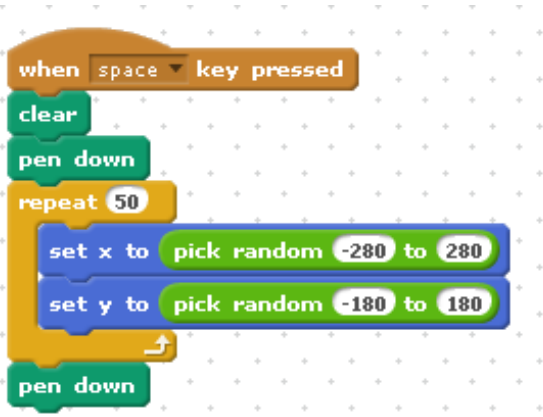


Fig. 5.43 Script ce realizează un desen aleator

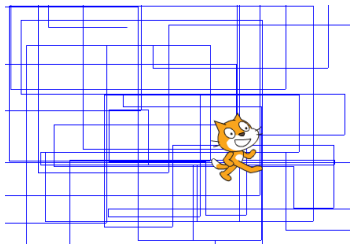


Fig.5.44 Rezultatul scriptului din Fig. 5.43

Program

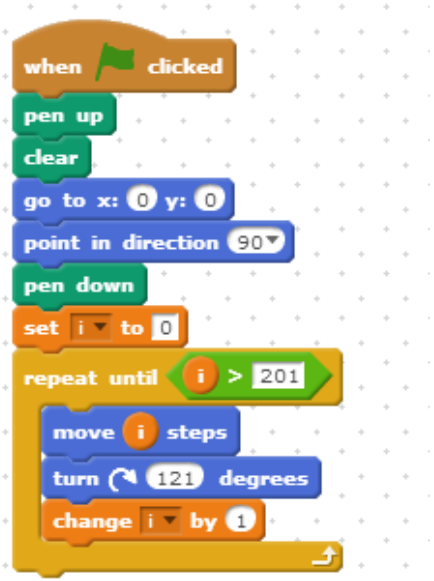


Fig. 5.45 Realizarea unui desen

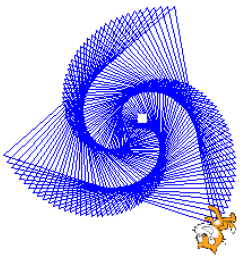


Fig. 5.46 Rezultatul scriptului din Fig. 5.45

Program

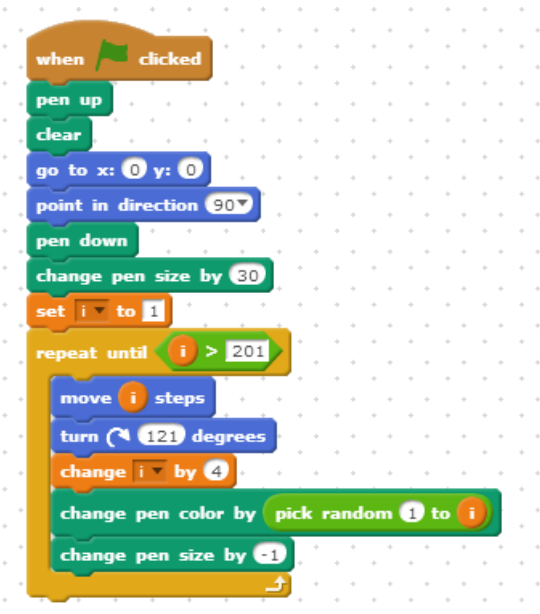


Fig. 5.47 Realizarea unui desen

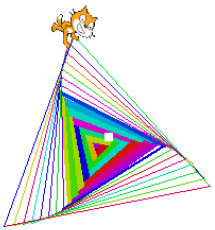


Fig. 5.48

Rezultatul scriptului din Fig. 5.47

Program

Pentru realizarea următorului program, aveți nevoie de un personaj cu mai multe costume – minim două.

In zona de script veți introduce blocurile din Fig. 5.49.

Rezultatul … rămâne SURPRIZĂ!

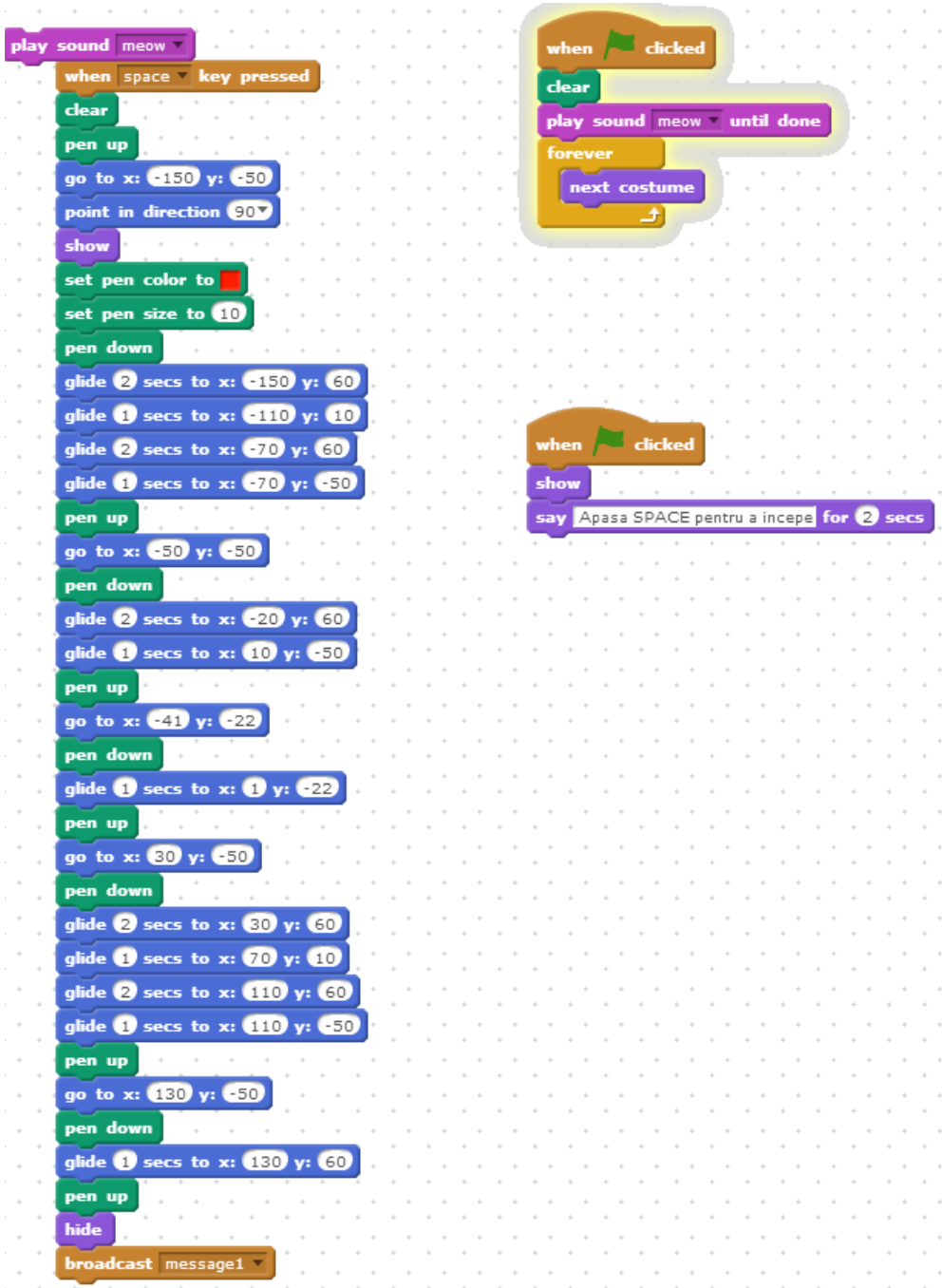


Fig. 5.49 O surpriză de Crăciun

5.11 Teme de lucru pentru ... voi

Împreună cu prietenii mei, vă propun să rezolvați câteva probleme.

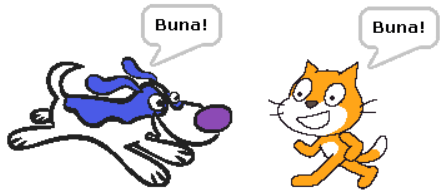
Să nu copiați de la rezolvări! Întâi vă gândiți și pe urmă vă puteți verfica!!!

Problema 1

Pată vrea să spună "Bună ziua" - îl puteți ajuta? Creați un bloc "Salut" și utilizați-l pentru a-l ajuta pe cățel.

Problema 2

Pată (personajul 1) şi Scratchy (personajul 2) ar dori să-şi spună "Bună". Îi puteți ajuta utilizând blocul "Salut" de două ori?

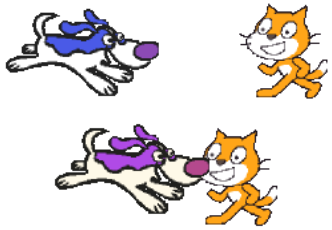


Problema 3

Îl puteți ajuta pe Pată să ajungă la copac?

Problema 4

Îl puteți ajuta pe Pată (personajul1) să ajungă la Scratchy (personajul2) și să-și schimbe culoarea când ajunge acolo?



Problema 5

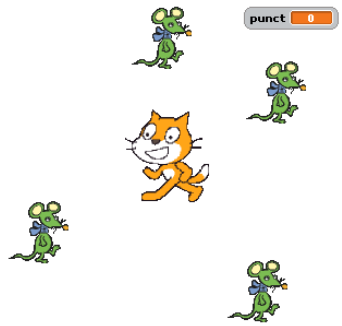
Îl puteți ajuta pe Pată să se prezinte atunci când se dă clic pe el? Nu uitaţi să daţi clic pe el, dar să nu apăsați steguleţul verde.

Problema 6

Îl puteți ajuta pe Scratchy să se miște împrejur folosind tastele săgeți sus/ jos/ dreapta/ stânga pentru a prinde toți șoarecii extratereștri?

Pentru fiecare șoarece, Scratchy va primi un punct.

Grăbiți-vă ... altfel șoarecii extratereștri vă vor mânca toată brânza!



Problema 7

Blocul 

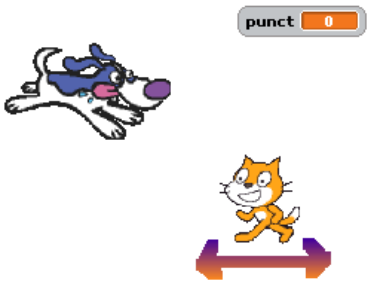
vă permite să rulaţi cod continuu. Puteți atașa blocuri pentru a-l ajuta pe Scratchy să se miște la dreapta şi la stânga în mod repetat?

Problema 8

Pată și Scratchy se joacă de-a "Prinde Mâța".

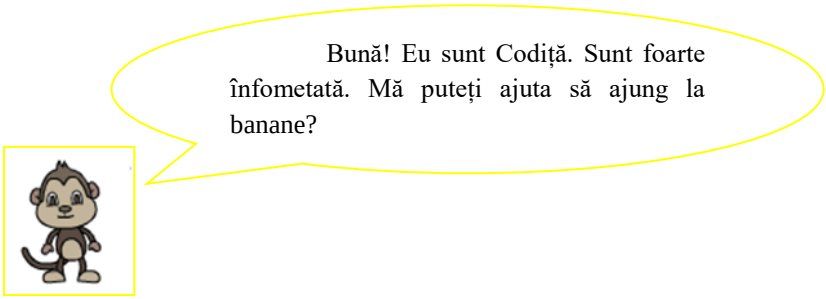
Scratchy se mișcă dreapta-stânga, iar cățelul trebuie ajutat, cu săgețile, să se miște.

Adaugă un bloc pentru a marca un punct și un bloc pentru a reda un sunet când Pată atinge pisoiul.



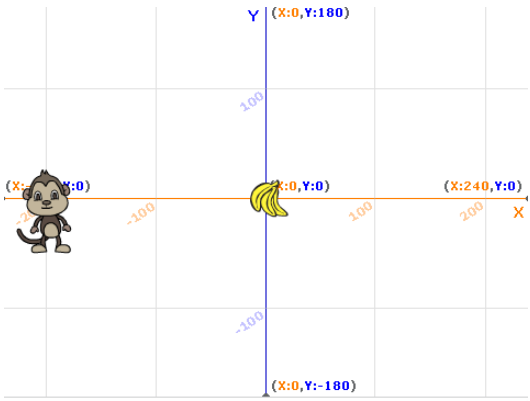
Problema 9

Dar Scratchy este o mâță specială: ea vine din Ținutul Programării! De aceea, Scratchy poate să își pună un scut de protecție. Atunci când Pată atinge mâța cu scut de protecție, acesta pierde un punct. Dacă totuși reușește să adune 5 puncte, Pată câștigă. Dacă Pată ajunge cu punctajul la 0, atunci Scratchy câștigă.



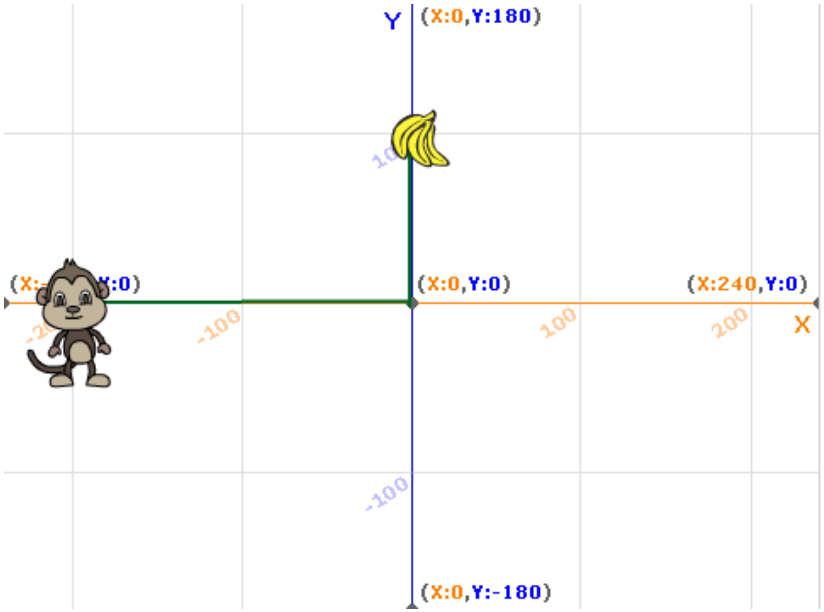
Problema 10

Ajutați maimuța să ajungă la banane.



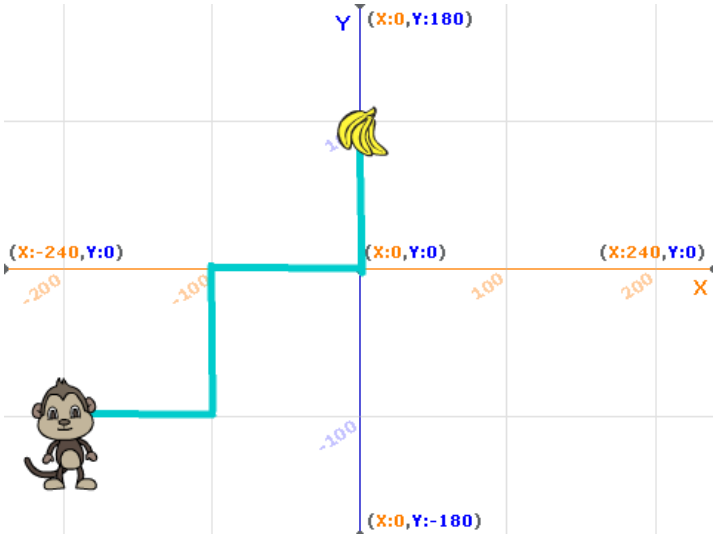
Problema 11

Scrieți linii de cod astfel încât să ajutați maimuța să ajungă la banane, urmărind cărarea verde.



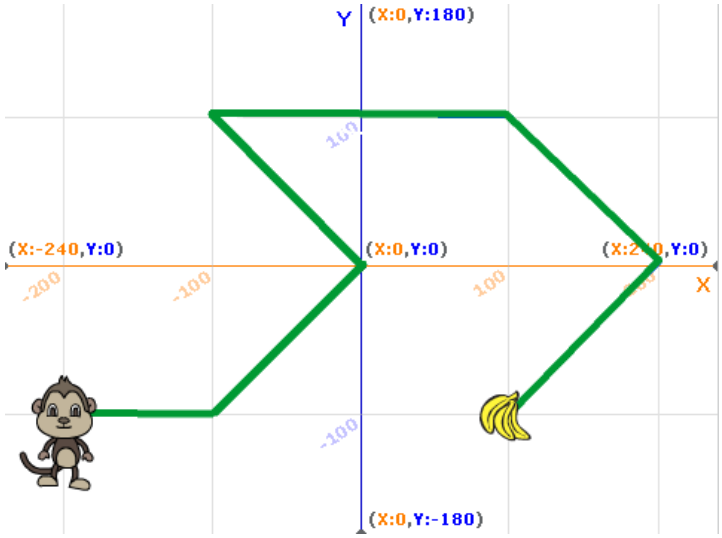
Problema 12

Scrieți linii de cod astfel încât să-l ajutați pe Codiță să ajungă la banane, urmărind cărarea albastră.



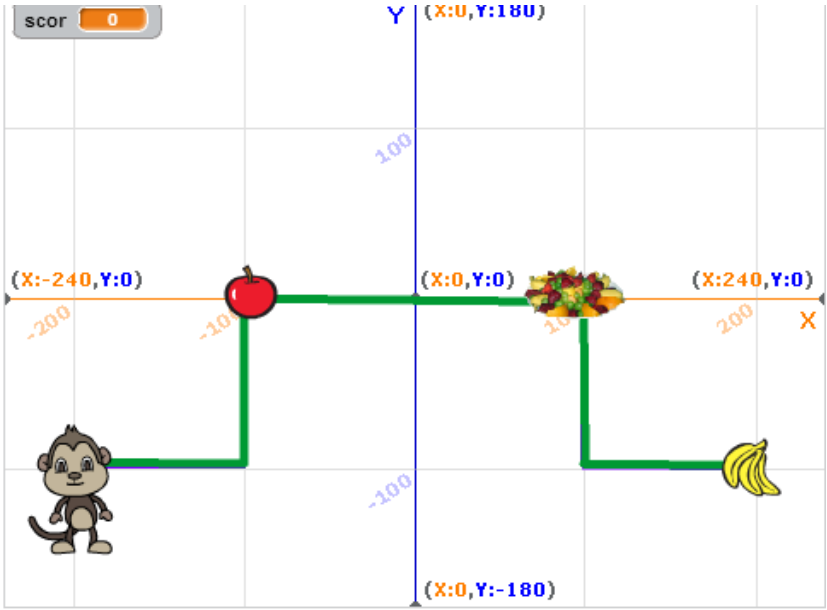
Problema 13

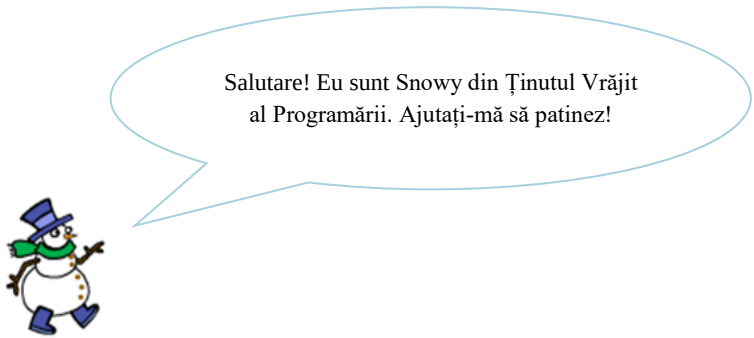
Scrieți linii de cod astfel încât să-l ajutați pe Codiță să ajungă la banane, urmărind cărarea verde.



Problema 14

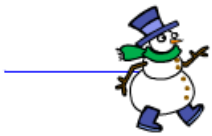
Pe cărarea verde, Codiță poate găsi foarte multe bunătăți. Scrieți linii de cod astfel încât să ajutați maimuța să strângă toată mâncarea. Atunci când se atinge de mâncare scorul va crește.





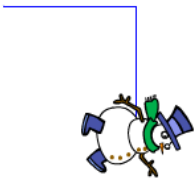
Problema 15

Ajutați-mă să creez o singură linie.



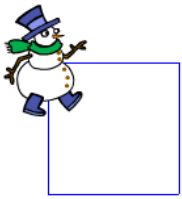
Problema 16

Acum să vedem dacă putem crea două linii care formează un unghi de 90 de grade.



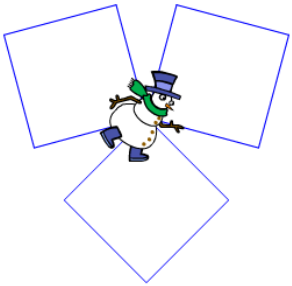
Problema 17

Se pare că am parcurs jumătate de drum în crearea unui pătrat. Să punem 4 linii împreună pentru a crea un pătrat.



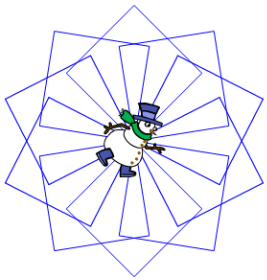
Problema 18

Haideți să creăm trei pătrate, rotind după fiecare pătrat. Asigurați-vă că rotiți cu 120 de grade înainte de fiecare pătrat nou.



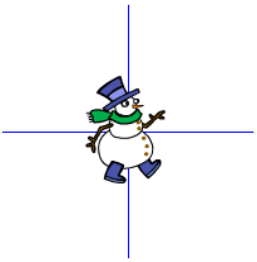
Problema 19

Puteți să creați un fulg de nea folosind blocul ”Repetă” pentru a face un pătrat de 10 ori și blocul ”Rotește” să rotească 36 de grade între pătrate?



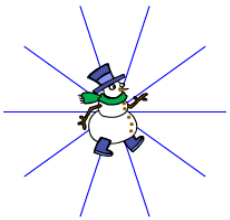
Problema 20

Utilizaţi blocul "Repetă" pentru a crea un semn plus. Ați observat că Snowy se poate muta înainte şi înapoi?



Problema 21

Acum, încercaţi să îl repetați de 10 ori. Câte grade trebuie să rotiți după fiecare linie?



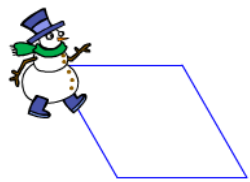
Problema 22

Haideți să îl repetăm de 90 de ori! De câte ori intră 90 în 360?

Sugestie: Este un număr chiar foarte mic.

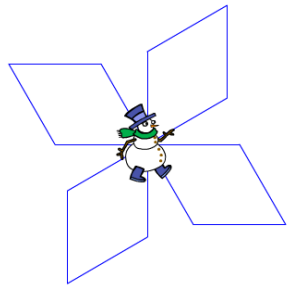
Problema 23

Haideți să creăm un paralelogram. Este chiar ca un pătrat dar are unghiuri diferite: unghiuri de 60 și 120 de grade în loc de unghiuri de 90 de grade.



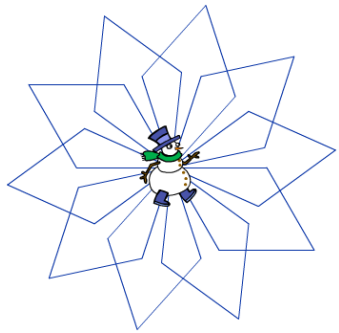
Problema 24

Ştiaţi că fiecare fulg de nea are o formă diferită? Haideți să creăm un nou fulg de nea utilizând blocul "Repetă" pentru a repeta un paralelogram de 4 ori, rotind la dreapta 90 de grade după fiecare paralelogram.



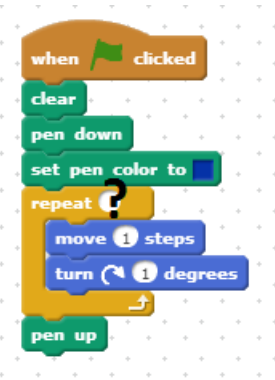
Problema 25

Acum, haideți să desenăam un nou fulg de zăpadă utilizând blocul de repetare pentru a desena un paralelogram de 10 ori, rotind la dreapta 36 de grade după fiecare dintre ele.



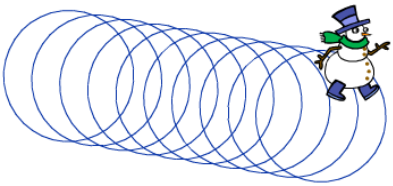
Problema 26

Un cerc este o formă specială. Puteți să vă dați seama cu ce număr să înlocuim semnul de întrebare pentru a desena un cerc?



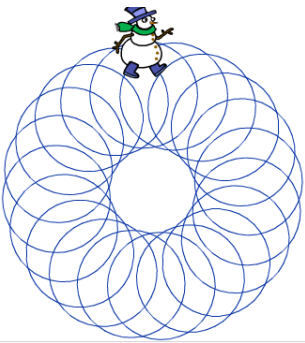
Problema 27

Utilizaţi noul bloc "Creare cerc" pentru a crea 10 cercuri suprapuse. Nu uitaţi să avansați între cercuri.



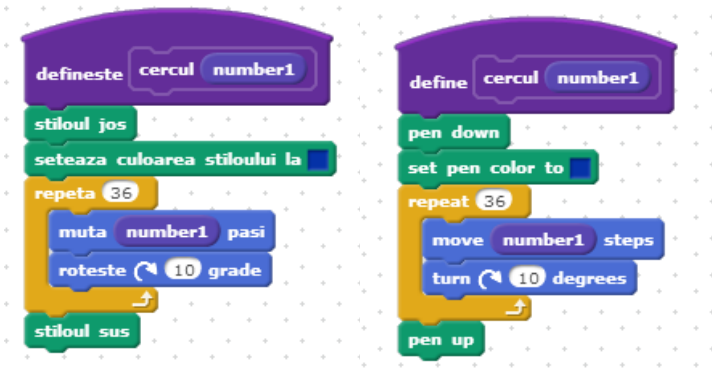
Problema 28

Acum haideți să creăm 20 de cercuri suprapuse, rotind 18 grade după fiecare cerc.



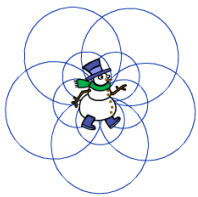
Problema 29

Mai jos este definit blocul "**cercul**", care poate face cercuri de diferite dimensiuni. Îl puteţi folosi pentru a crea un mic cerc de dimensiunea 5 şi un cerc mai mare de dimensiune 10?



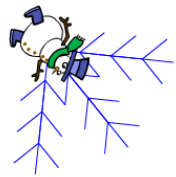
Problema 30

Se pot crea modele complicate de zăpadă cu forme foarte simple. Puteți face un model repetând 5 cercuri de dimensiune 5 și 5 cercuri de dimensiune 10?



Problema 31

Încercaţi să utilizaţi blocul "Fulg", definit mai jos, pentru a crea trei ramuri, care încep să arate ca un fulg de nea.



Problema 32

Acum haideți să îl repetăm de 8 ori pentru a face un frumos fulg de nea!

O ultimă problemă .....

Pată și Scratchy au învățat să programeze! Un prilej bun pentru a da o petrecere. Ei și-au invitat prietenii.

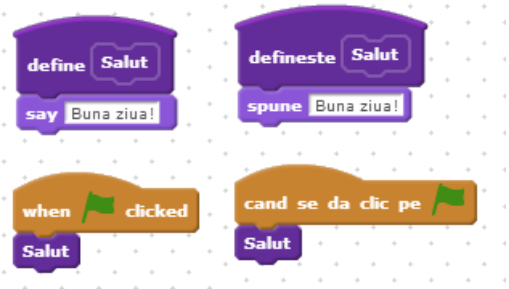
Ei te roagă să găsești un fundal potrivit pentru asta.

Tu fă-le o surpriză: pune-le muzică și fă-i să danseze!

Ei se vor bucura!!!

5.12 Rezolvarea problemelor propuse

Problema 1



Problema 2

Indicație

Se va folosi blocul de la problema 1, pentru fiecare dintre cele două personaje.

Problema 3

Problema 4

Indicație

Se va modifica scriptul de la problema 3, prin adaugarea următoarelor blocuri

Problema 5

Problema 6

Atenție!

Scriptul corespunde personajului Scratchy.


when clicked
set punct to 0
forever
if key up arrow pressed? then
change y by 10
if key down arrow pressed? then
change y by -10
if key right arrow pressed? then
change x by 10
if key left arrow pressed? then
change x by -10
atinge
define punct
change punct by 1


define atinge
if touching Mouse1 ? then
go to x: 0 y: 0
punct
if touching Mouse2 ? then
go to x: 0 y: 0
punct
if touching Mouse3 ? then
go to x: 0 y: 0
punct
if touching Mouse4 ? then
go to x: 0 y: 0
punct


cand se da clic pe
seteaza punct la 0
la infinit
dacă tasta sageata sus este apasata? atunci
modifica y cu 10
dacă tasta sageata jos este apasata? atunci
modifica y cu -10
dacă tasta sageata dreapta este apasata? atunci
modifica x cu 10
dacă tasta sageata stanga este apasata? atunci
modifica x cu -10
atinge
defineste punct
modifica punct cu 1

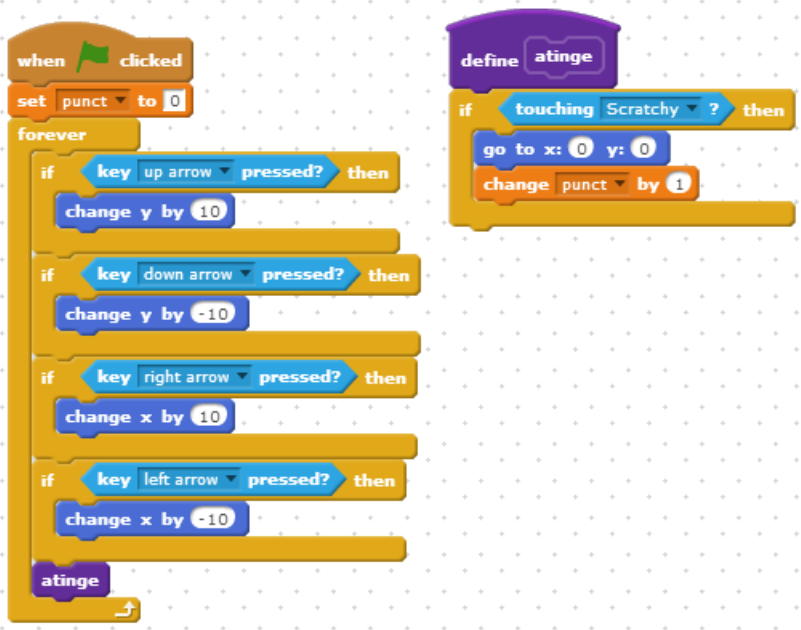

defineste atinge
dacă touching Mouse1 ? atunci
du-te la x: 0 y: 0
punct
dacă touching Mouse2 ? atunci
du-te la x: 0 y: 0
punct
dacă touching Mouse3 ? atunci
du-te la x: 0 y: 0
punct
dacă touching Mouse4 ? atunci
du-te la x: 0 y: 0
punct

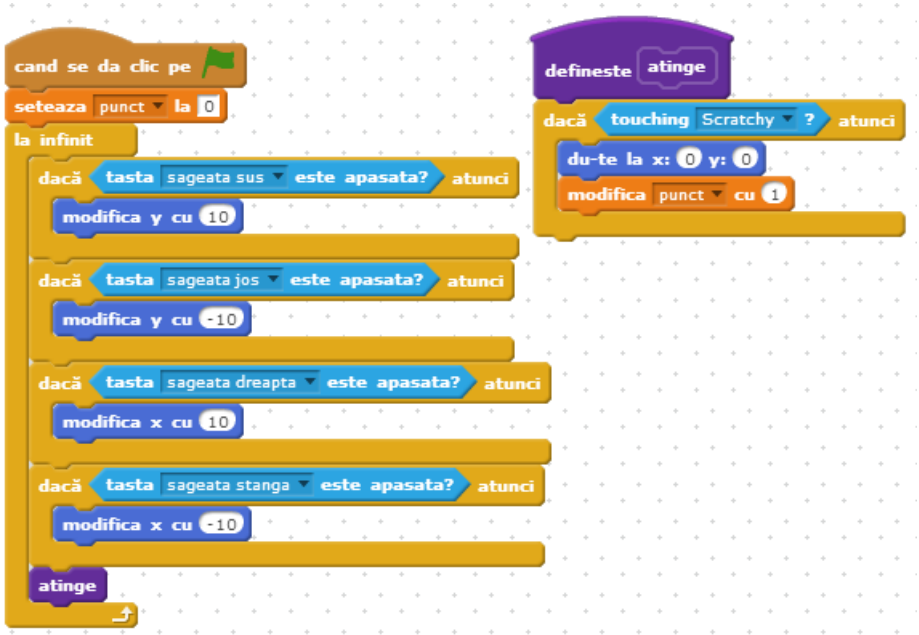
Problema 7

Problema 8

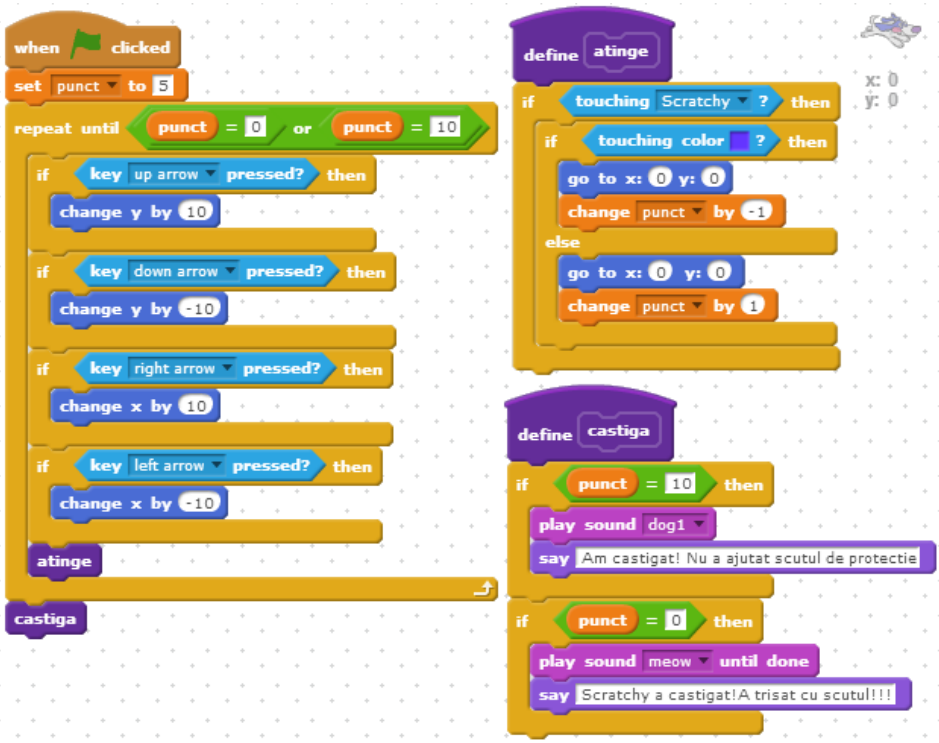
Atenție!

Pentru personajul1 (Scratchy) se scrie scriptul de la problema 7. Pentru personajul2 (Pată) se scrie scriptul de mai jos.



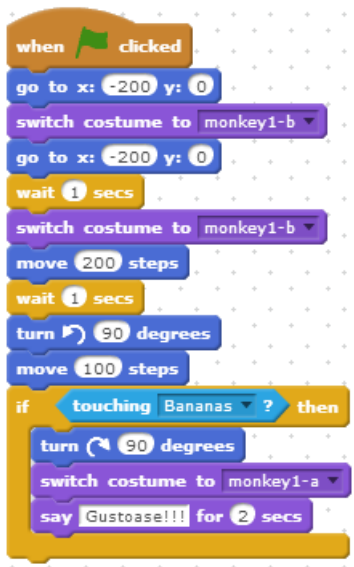

cand se da clic pe
seteaza punct la 0
la infinit
dacă tasta sageata sus este apasata? atunci
modifica y cu 10
dacă tasta sageata jos este apasata? atunci
modifica y cu -10
dacă tasta sageata dreapta este apasata? atunci
modifica x cu 10
dacă tasta sageata stanga este apasata? atunci
modifica x cu -10
atinge
defineste atinge
dacă touching Scratchy ? atunci
du-te la x: 0 y: 0
modifica punct cu 1

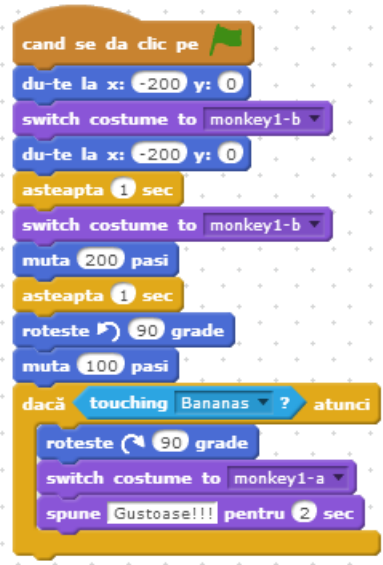
Problema 9



Problema 10

Problema 11


when clicked
go to x: -200 y: 0
switch costume to monkey1-b
go to x: -200 y: 0
wait 1 secs
switch costume to monkey1-b
move 200 steps
wait 1 secs
turn 90 degrees
move 100 steps
if touching Bananas ? then
turn 90 degrees
switch costume to monkey1-a
say Gustoase!!! for 2 secs


cand se da clic pe
du-te la x: -200 y: 0
switch costume to monkey1-b
du-te la x: -200 y: 0
asteapta 1 sec
switch costume to monkey1-b
muta 200 pasi
asteapta 1 sec
roteste 90 grade
muta 100 pasi
dacă touching Bananas ? atunci
roteste 90 grade
switch costume to monkey1-a
spune Gustoase!!! pentru 2 sec

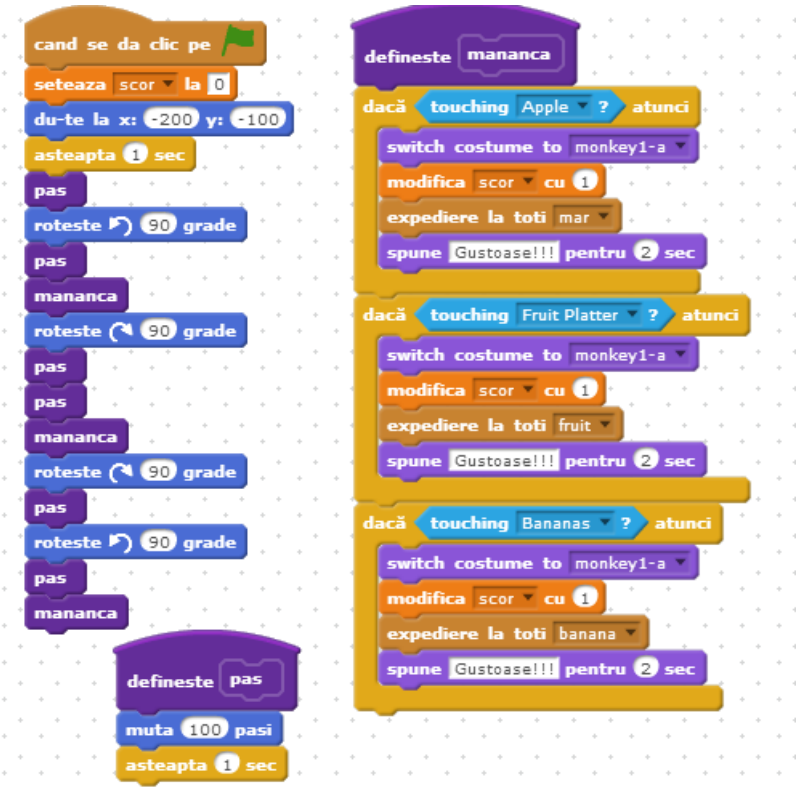
Problema 12


when clicked
go to x: -200 y: -100
wait 1 secs
move 100 steps
wait 1 secs
turn 90 degrees
move 100 steps
turn 90 degrees
wait 1 secs
move 100 steps
turn 90 degrees
wait 1 secs
move 100 steps
if touching Bananas ? then
turn 90 degrees
switch costume to monkey1-a
say Gustoase!!! for 2 secs
cand se da clic pe
du-te la x: -200 y: -100
asteapta 1 sec
muta 100 pasi
asteapta 1 sec
roteste 90 grade
muta 100 pasi
roteste 90 grade
asteapta 1 sec
muta 100 pasi
roteste 90 grade
asteapta 1 sec
muta 100 pasi
dacă touching Bananas ? atunci
roteste 90 grade
switch costume to monkey1-a
spune Gustoase!!! pentru 2 sec

Problema 13


when clicked
go to x: -200 y: -100
wait 1 secs
pas
turn 45 degrees
pasD
turn 90 degrees
pasD
turn 125 degrees
pas
pas
turn 45 degrees
pasD
turn 90 degrees
pasD
turn 125 degrees
mananca
define mananca
if touching Bananas ? then
switch costume to monkey1-a
broadcast banane
say Gustoase!!! for 2 secs
define pas
move 100 steps
wait 1 secs
define pasD
move 142 steps
wait 1 secs

Problema 14


cand se da clic pe
seteaza scor la 0
du-te la x: -200 y: -100
asteapta 1 sec
pas
roteste 90 grade
pas
mananca
roteste 90 grade
pas
pas
mananca
roteste 90 grade
pas
roteste 90 grade
pas
mananca
defineste pas
muta 100 pasi
asteapta 1 sec
defineste mananca
dacă touching Apple ? atunci
switch costume to monkey1-a
modifica scor cu 1
expediere la toti mar
spune Gustoase!!! pentru 2 sec
dacă touching Fruit Platter ? atunci
switch costume to monkey1-a
modifica scor cu 1
expediere la toti fruit
spune Gustoase!!! pentru 2 sec
dacă touching Bananas ? atunci
switch costume to monkey1-a
modifica scor cu 1
expediere la toti banana
spune Gustoase!!! pentru 2 sec

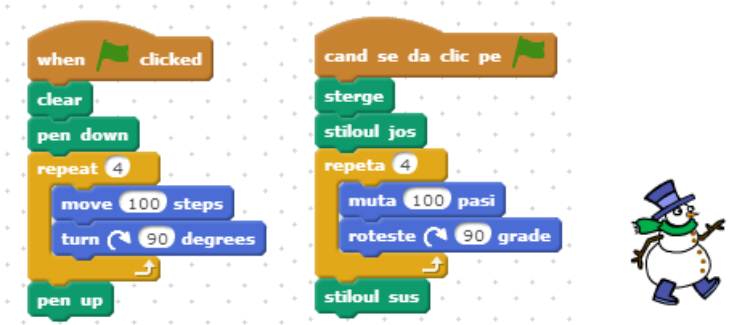

when clicked
set scor to 0
go to x: -200 y: -100
wait 1 secs
pas
turn 90 degrees
pas
mananca
turn 90 degrees
pas
pas
mananca
turn 90 degrees
pas
turn 90 degrees
pas
mananca
define pas
move 100 steps
wait 1 secs
define mananca
if touching Apple ? then
switch costume to monkey1-a
change scor by 1
broadcast mar
say Gustoase!!! for 2 secs
if touching Fruit Platter ? then
switch costume to monkey1-a
change scor by 1
broadcast fruit
say Gustoase!!! for 2 secs
if touching Bananas ? then
switch costume to monkey1-a
change scor by 1
broadcast banana
say Gustoase!!! for 2 secs

Problema 15

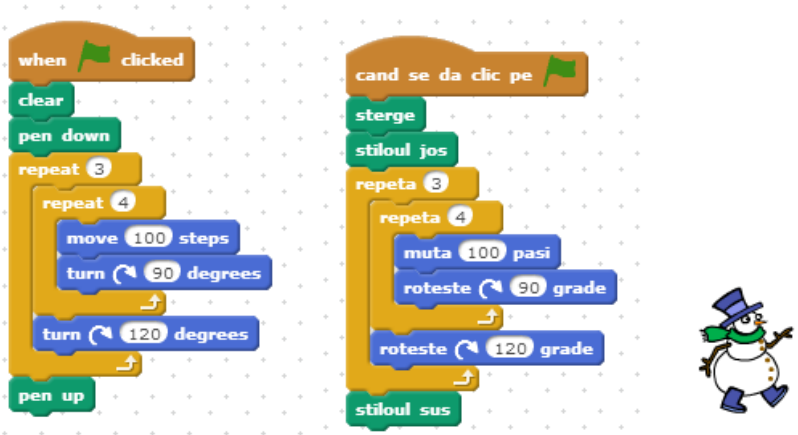

when clicked
clear
pen down
move 100 steps
pen up
cand se da clic pe
sterge
stiloul jos
muta 100 pasi
stiloul sus

Problema 16

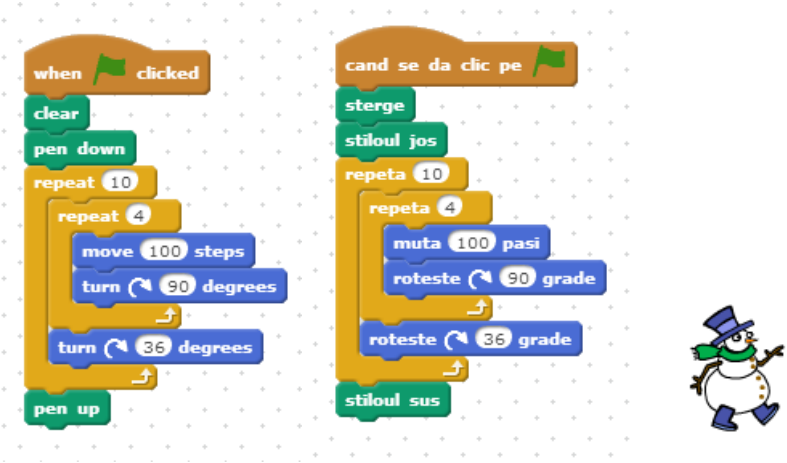
Problema 17



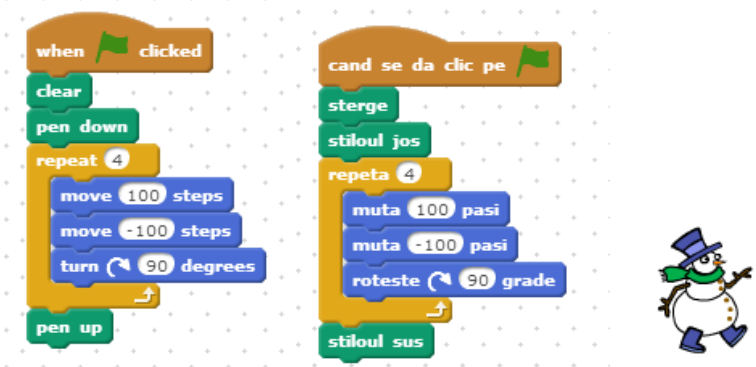
Problema 18



Problema 19



*Problema 20

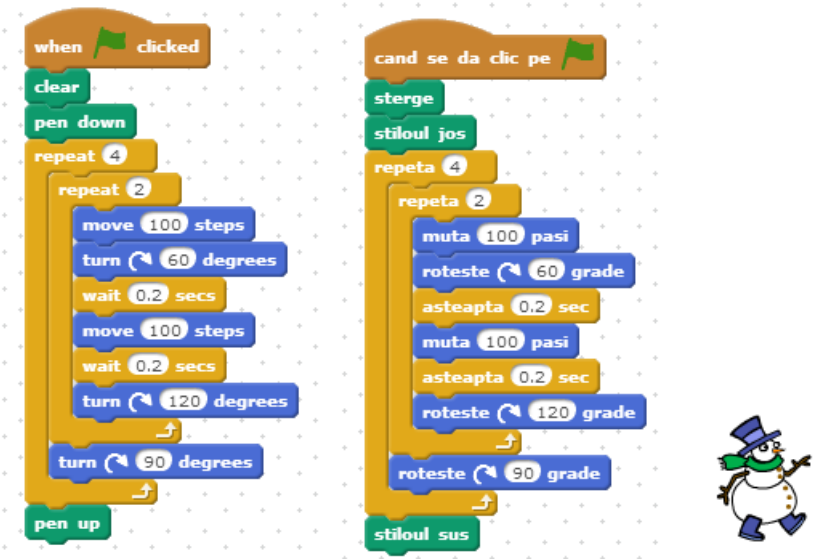


Problema 21

Problema 22

Problema 23

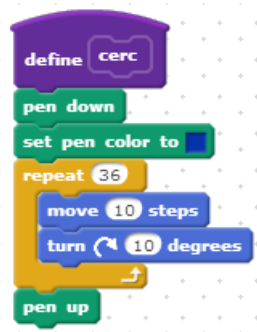
Problema 24



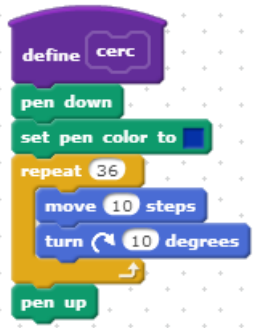
Problema 25

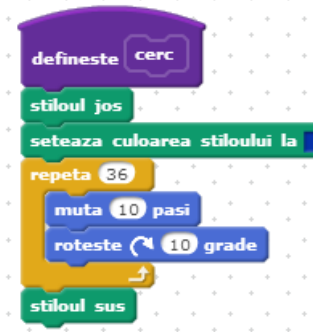
Problema 26

Problema 27

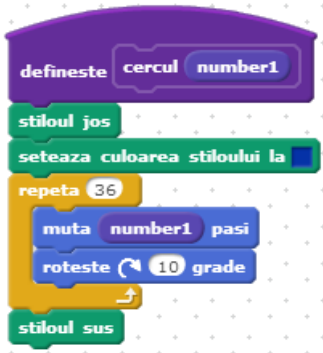


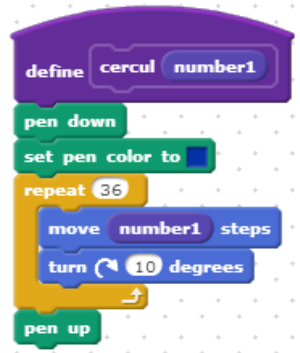
Problema 28

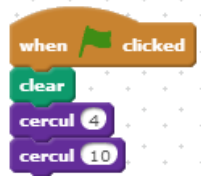




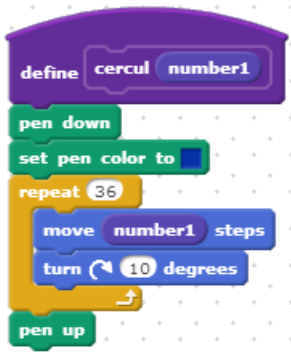
Problema 29

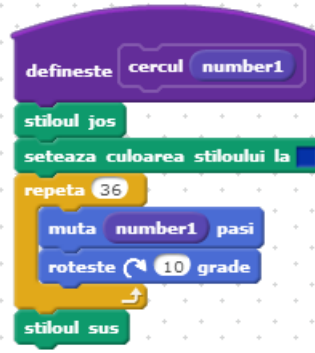


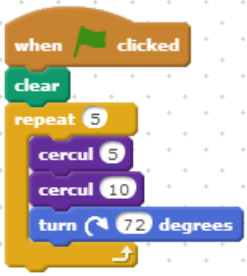




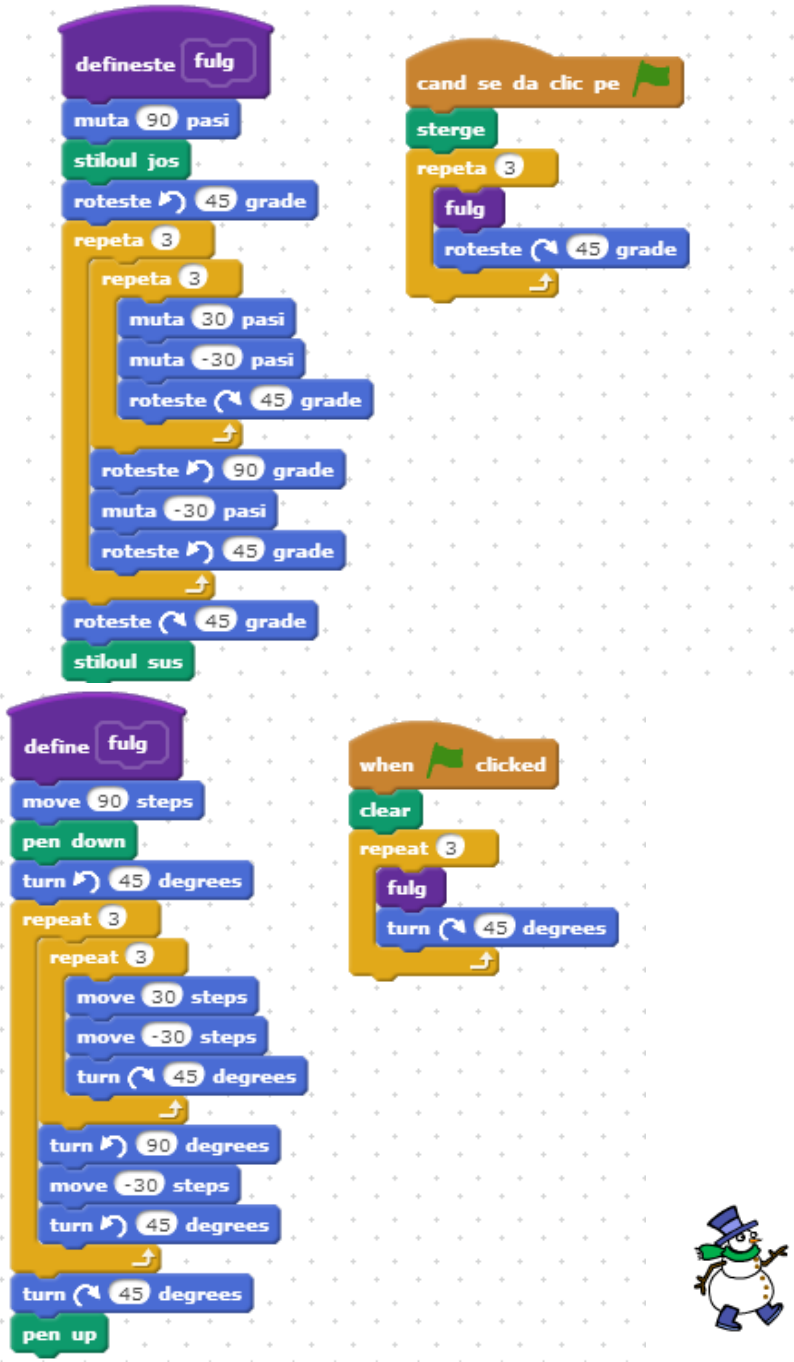
Problema 30







Problema 31


defineste fulg
muta 90 pasi
stiloul jos
roteste 45 grade
repeta 3
repeta 3
muta 30 pasi
muta -30 pasi
roteste 45 grade
roteste 90 grade
muta -30 pasi
roteste 45 grade
roteste 45 grade
stiloul sus
cand se da clic pe
sterge
repeta 3
fulg
roteste 45 grade
define fulg
move 90 steps
pen down
turn 45 degrees
repeat 3
repeat 3
move 30 steps
move -30 steps
turn 45 degrees
turn 90 degrees
move -30 steps
turn 45 degrees
turn 45 degrees
pen up
when clicked
clear
repeat 3
fulg
turn 45 degrees

Problema 32


when clicked
clear
repeat 8
fulg
turn 45 degrees
define fulg
move 90 steps
pen down
turn 45 degrees
repeat 3
repeat 3
move 30 steps
move -30 steps
turn 45 degrees
turn 90 degrees
move -30 steps
turn 45 degrees
turn 45 degrees
pen up
cand se da clic pe
sterge
repeta 8
fulg
roteste 45 grade
defineste fulg
muta 90 pasi
stiloul jos
roteste 45 grade
repeta 3
repeta 3
muta 30 pasi
muta -30 pasi
roteste 45 grade
roteste 90 grade
muta -30 pasi
roteste 45 grade
roteste 45 grade
stiloul sus

5.13 Algoritmi elementari implementați în Scratch

Problemă

Se dă un număr natural n. Să se spună dacă numărul este/ nu este pătrat perfect.

Fig. 5.50 Verificarea unui număr dacă este pătrat perfect

Problemă

Se dă un număr natural n. Să se afișeze primele n numere pătrate perfecte.

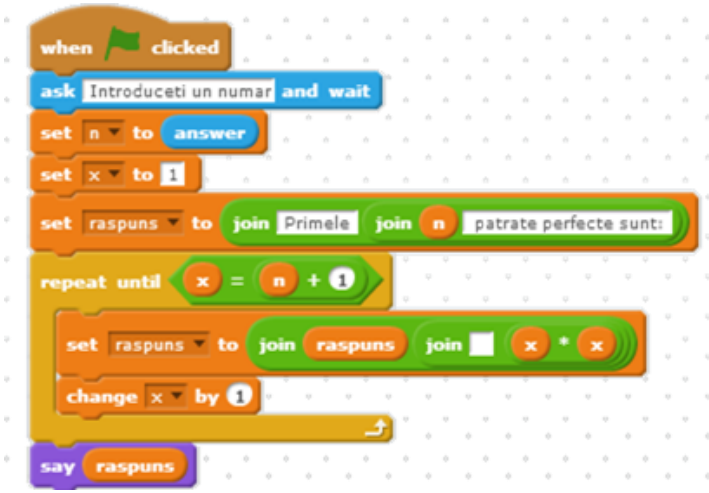


Fig. 5.51 Afișarea primelor "n" pătrate perfecte

Problemă

Se dă un număr natural n. Să se spună dacă numărul este sau nu este prim.

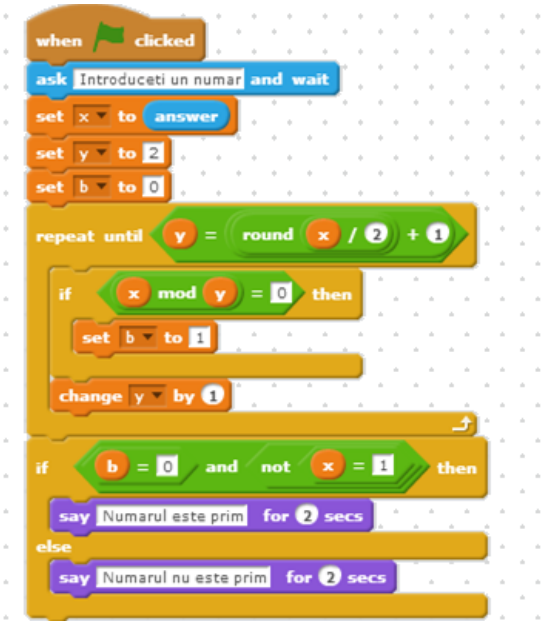


Fig. 5.52Verificarea unui număr dacă este prim

Problemă

Un număr este palindrom daca este egal cu inversul sau. (Ex. 121 este palindrom).

Se dă un număr natural n. Să se spună dacă numărul este sau nu este palindrom.

Fig. 5.53 Verificarea unui numătr dacă este palindrom

Problemă

De la tastatură se introduce un număr natural. Să se spună care este cel mai mare număr care se poate forma din cifrele sale.

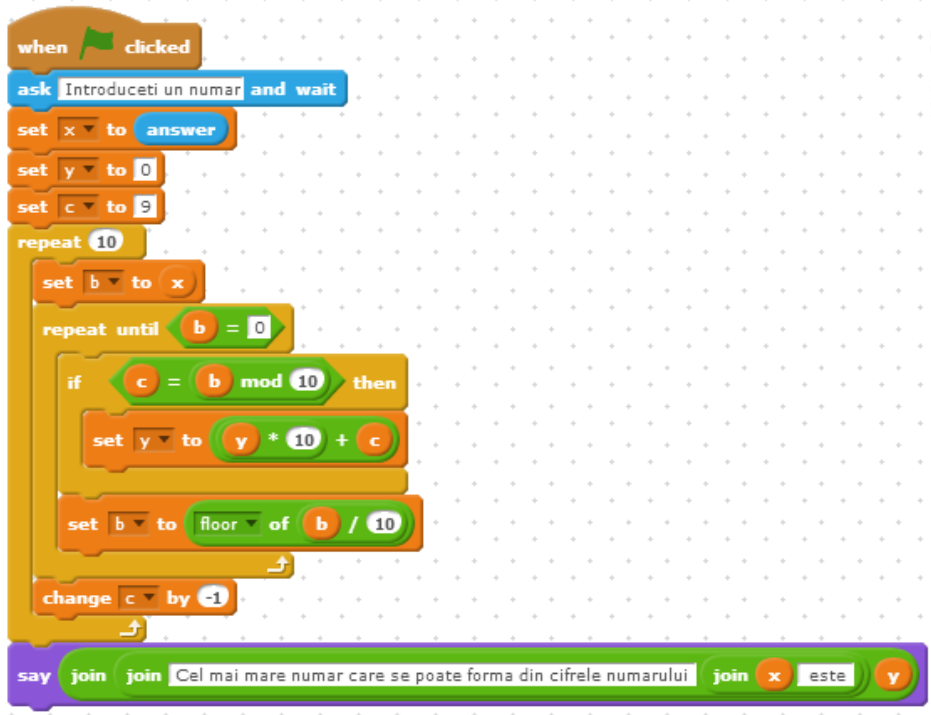


Fig. 5.54 Afişarea celui mai mare număr care se poate forma din cifrele unui număr

Problemă

De la tastatură se introduce un număr natural n. Să se afișeze primele n numere prime.

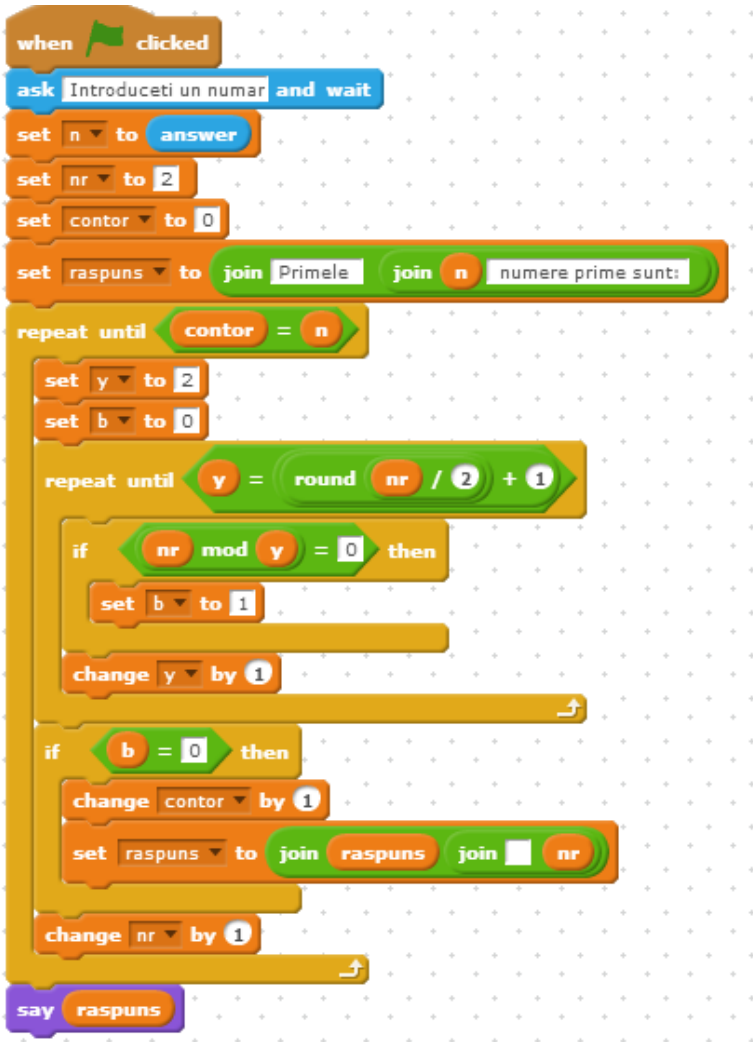


Fig. 5.55 Afișarea primelor "n" numere prime

Capitolul 6. **Raspberry PI**

Eu i-am spus zmeurică de teleportare. Mă puteți scoate din calculator și mă puteți robotiza.

6.1 Ce este Raspberry PI

Conform Wikipedia, Raspberry PI este un calculator, de dimensiunea unei cărți de credit, dezvoltat de o fundație britanică cu scopul declarat de a facilita însușirea bazelor informaticii în sistemul de învățământ primar și secundar. Este dotat cu un procesor ARM11 cu o frecvență de 700MHz, o memorie RAM de 256MB în varianta A și 512MB în varianta B. În plus este dotat cu un set de pini GPIO care permit realizarea de conexiuni de intrare/ieșire cu alte dispozitive (senzori, butoane, leduri etc).

Popularitatea acestui mini-calculator a crescut fulminant încă de la lansarea din 2012. Prețul scăzut, dimensiunile mici, sistemul de operare Linux cu mii de proiecte open-source, posibilitatea de a realiza proiecte de automatizare cu buget scăzut și, nu în ultimul rând, o comunitate mare de entuziaști au contribuit în egală măsură la notorietatea acestei platforme.

Sistemele de operare care pot rula pe Raspberry PI sunt bazate majoritar pe kernel Linux. Ca o noutate, echipamentele de generație 2 permit instalarea sistemului de operare Windows 10.

Principalele sisteme de operare care pot fi instalate sunt următoarele:

- Raspbian (recomandat)
- Archlinux ARM
- Raspbmc

- OpenELEC
- RISC OS
- Pidora

Pentru scopul lucrării de față vom exemplifica crearea de aplicații Scratch care vor utiliza pinii GPIO. Sistemul de operare utilizat este Raspbian.

Dacă adăugăm la un Raspberry Pi un card de memorie SD (cum sunt cele din aparatele de fotografiat), un alimentator microUSB (majoritatea telefoanelor sau tabletelor folosesc unul), o tastatură USB, un mouse USB și un monitor cu intrare HDMI avem un calculator gata pregătit pentru a ne conduce în minunata lume a programării și automatizărilor.

6.2 Prezentare versiuni

La momentul redactării lucrării de față sunt comercializate cinci versiuni A, A+, B, B+ si generația 2 Model B. În plus, există și un modul computațional, a cărui prezentare depășește scopul acestei cărți.

Tabel 6-1Prezentare comparativă a versiunilor de Raspberry PI

	Generatia 1 Model A	Generatia 1 Model A+	Generatia 1 Model B	Generatia 1 Model B+	Generatia 2 Model B
Memorie RAM	256MB	256MB	512MB	512MB	1GB
Procesor	ARM11	ARM11	ARM11	ARM11	ARM Cortex-A7
Frecvență	700MHz	700MHz	700MHz	700MHz	900MHz
Tip alimentare	MicroUSB	MicroUSB	MicroUSB	MicroUSB	MicroUSB
Putere	1.5W	1W	3.5W	3W	4W

consum	=5Vx0.3A	=5Vx0.2A	=5Vx0.7A	=5Vx0.6A	=5Vx0.8A
USB	1	1	2	4	4
Header GPIO	26 pini din care 17 GPIO	40 pini din care 28 GPIO	26 pini din care 17 GPIO	40 pini din care 28 GPIO	40 pini din care 28 GPIO
Ieșire video RCA	Da	Nu	Da	Nu	Nu
Ieșire video HDMI	Da	Da	Da	Da	Da
Ieșire audio jack stereo 3.5mm	Da	Da	Da	Da	Da
Rețea	Nu	Nu	1xRJ45	1xRJ45	1xRJ45
Slot card memorie	SD/ MMC/ SDIO	MicroSD	SD/ MMC/ SDIO	MicroSD	MicroSD

Fig. 6.1Raspberry PI Model A

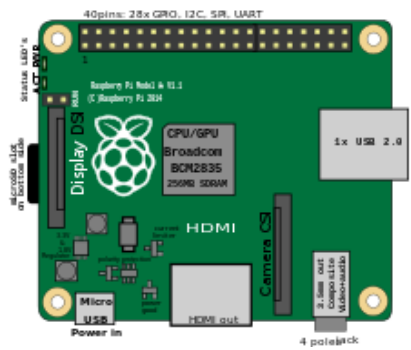


Fig. 6.2 Raspberry PI Model A+

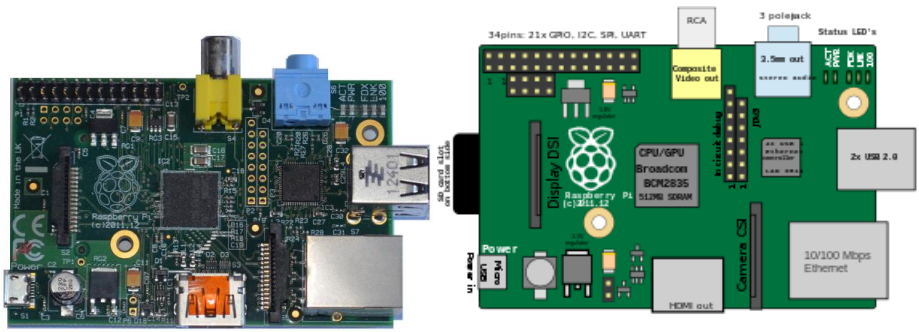


Fig. 6.3 Raspberry PI Model B

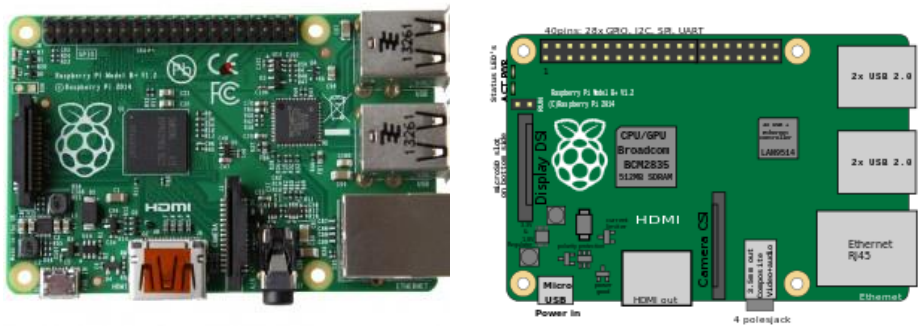


Fig. 6.4 Raspberry PI Model B+si Raspberry PI 2 model B

6.3 Pinii GPIO

Pinii GPIO sunt comoara pe care Raspberry PI o pune la dispoziție celor mai tenace pionieri prin tainele roboticii.

GPIO este termenul ce definește perifericul al cărui comportament de intrare sau ieșire, poate fi controlat de către utilizator.

Prin programarea ca intrări a acestor pini, putem primi informații de la senzori sau le putem defini rol de comutatoare (butoane) de comandă. Programați ca ieșiri, putem aprinde leduri

sau controla motoare. Lista de utilizări nu se oprește aici și poate fi continuată dacă înțelegem două concepte de bază:

- Intrare - Dacă dorim ca un eveniment din exterior să producă un eveniment în aplicațiile dezvoltate de noi cu Raspberry PI + Scratch, atunci trebuie sa căutăm un mod prin care să conectăm electric efectul evenimentului de un pin GPIO programat ca intrare. De exemplu, putem conecta un buton care, prin apăsare, să determine un mieunat al pisoiului Scratch.
- Ieșire - Dacă dorim ca un eveniment din aplicația Scratch să producă un eveniment în exteriorul Raspberry Pi, atunci trebuie să căutăm o modalitate prin care semnalul trimis de noi la un pin GPIO, programat ca ieșire, să genereze un eveniment extern. De exemplu, mângâierea pisoiului Scratch prin apăsarea butonului mouse-ului pe suprafața acestuia poate determina aprinderea unor leduri.

6.3.1 **GPIO pentru model A si model B**

Pinii GPIO (General Purpose Input/Output)

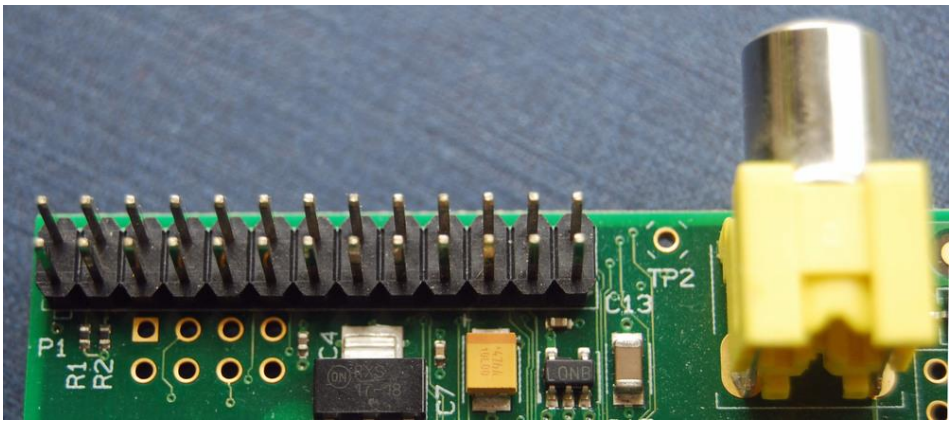


Fig. 6.5 Headerul GPIO

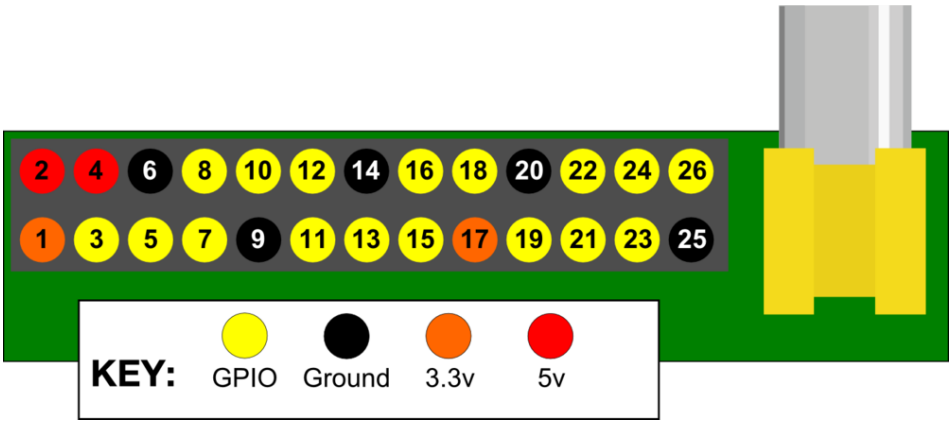


Fig. 6.6Numerotarea pinilor pe headerul GPIO

6.3.2 **GPIO pentru Modelul A+, B+ si Modelul 2B**

Raspberry Pi B+ J8 Header

Pin#	NAME	NAME	Pin#
01	3.3v DC Power	DC Power 5v	02
03	GPIO02 (SDA1 , I2C)	DC Power 5v	04
05	GPIO03 (SCL1 , I2C)	Ground	06
07	GPIO04 (GPIO_GCLK)	(TXD0) GPIO14	08
09	Ground	(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)	(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)	Ground	14
15	GPIO22 (GPIO_GEN3)	(GPIO_GEN4) GPIO23	16
17	3.3v DC Power	(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)	Ground	20
21	GPIO09 (SPI_MISO)	(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)	(SPI_CE0_N) GPIO08	24
25	Ground	(SPI_CE1_N) GPIO07	26
27	ID_SD (I2C ID EEPROM)	(I2C ID EEPROM) ID_SC	28
29	GPIO05	Ground	30
31	GPIO06	GPIO12	32
33	GPIO13	Ground	34
35	GPIO19	GPIO16	36
37	GPIO26	GPIO20	38
39	Ground	GPIO21	40

Rev. 1.1
16/07/2014

http://www.element14.com

6.4 Instalarea sistemului de operare Raspbian

Un ghid al modului de instalare se regăsește consultând pagina web http://www.raspberrypi.org/downloads/ . În continuare vom prezenta pe scurt o procedură de instalare pentru versiunea ”September 2014” a sistemului Raspbian bazat pe Debian Wheezy.

6.4.1 Descărcarea imaginii sistemului

Fișierul cu imaginea sistemului se găsește arhivat în format zip și trebuie descarcat de la adresa web

http://downloads.raspberrypi.org/raspbian_latest

6.4.2 Dezarhivare

Dezarhivați fișierul cu imaginea sistemului. Veți obține un fișier denumit ***2014-09-09-wheezy-raspbian.img***

> ***Observație**: 2014-09-09-wheezy-raspbian.img este versiunea de la data redactării lucrării de față. Important este să folosiți întotdeauna ultima versiune a sistemului de operare.*

6.4.3 Scrierea imaginii utilizând Windows

> ***Observație**: Dacă nu aveți la dispoziție un calculator cu sistem de operare Windows, pentru Linux și Mac OS consultați http://www.raspberrypi.org/documentation/installation/installing-images/README.md*

6.4.3.1 Necesar

- un card de memorie SD cu o capacitate de minim 4GB. Puteți folosi și carduri cu capacități mai mari. Important este sa utilizați un card de memorie cu o clasă de viteză mare (recomandabil 10 sau mai mare)
- un calculator cu sistem de operare Windows

- un cititor de carduri (daca nu aveți unul intern, așa cum au majoritatea laptopurilor, va trebui să achiziționați unul extern)

6.4.3.2 Pași

1) Inserați cardul SD în cititor și memorați litera de drive care i-a fost asociată (de exemplu **H:**).
2) Pentru scrierea imaginii .img pe un SD card, descărcați kitul de instalare Win32DiskImager de la adresa http://sourceforge.net/projects/win32diskimager/ . Instalați Win32DiskImager. După instalare rulați Win32DiskImager

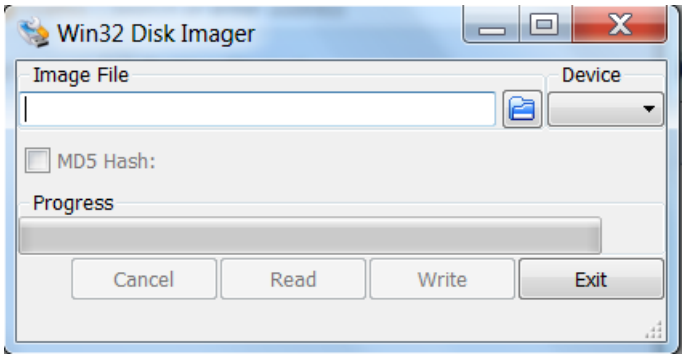


Fig. 6.7

3) Selectați fișierul 2014-09-09-wheezy-raspbian.img cu imaginea sistemului salvat la pasul 6.4.2

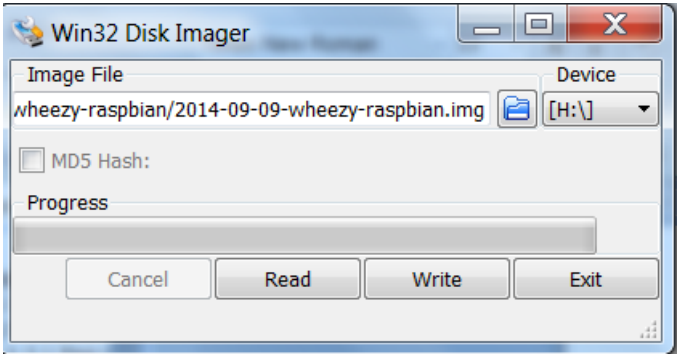


Fig. 6.8

***Observație**: Va trebui sa rulați Win32DiskImager cu drepturi de administrator, pentru a putea scrie pe cardul SD.*

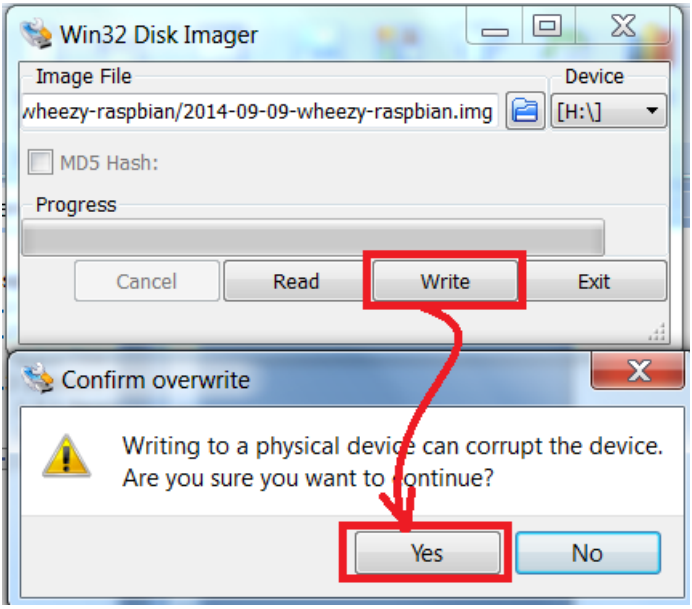


Fig. 6.9

Evoluția scrierii se poate urmări în bara de progres.

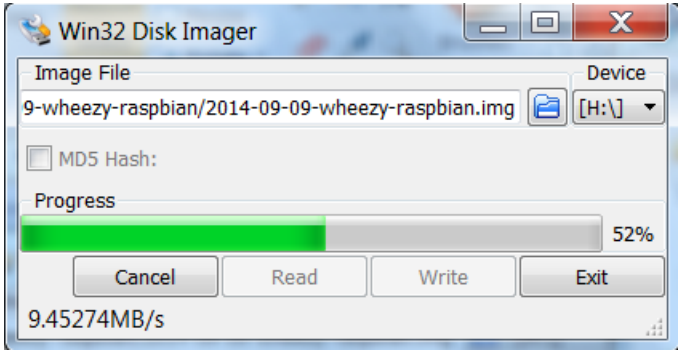


Fig. 6.10

Dacă nu au fost erori, după terminarea scrierii va fi afișat mesajul ”Write Successful”.

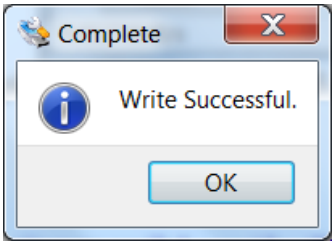


Fig. 6.11

4) Apăsați butonul Ok
5) In Windows Explorer executați clic dreapta pe litera drive-ului cardului SD (in exemplul nostru H:). Selectați Eject.

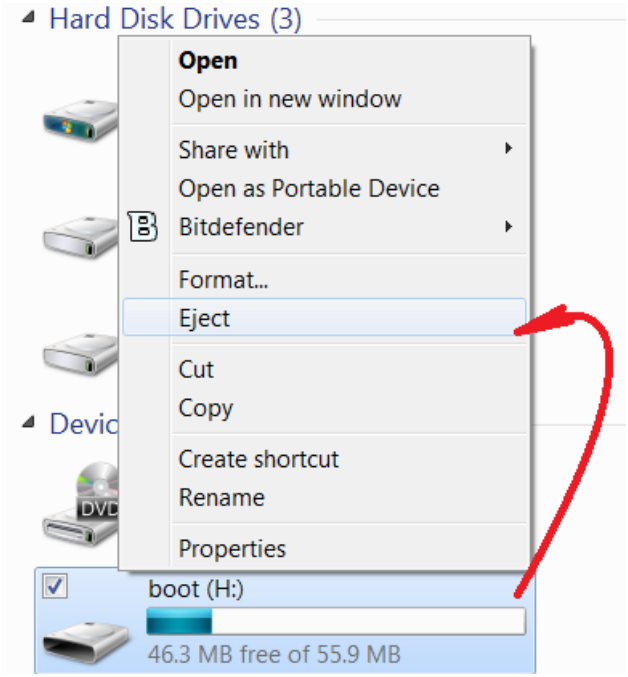


Fig. 6.12

Dacă procedura a fost efectuată corect va fi afișat mesajul ”Safe To Remove Hardware”.

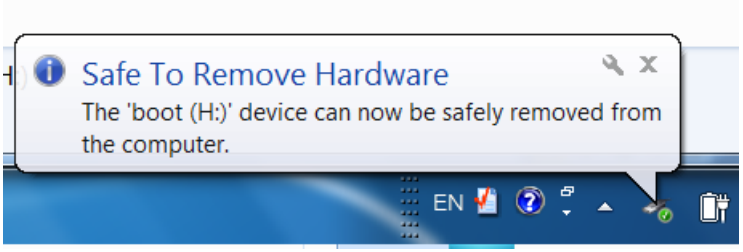


Fig. 6.13

6) Extrageți cardul de memorie din reader. Sunteți gata să porniți într-o nouă călătorie.

6.5 Prima rulare

La prima rulare va fi afișată o fereastră de dialog cu o unealtă de configurare în care putem opta pentru modul în care vom utiliza această fantastică jucărie.

Observații:

1. *Putem, oricând, să accesăm unealta de configurare prin apelarea, din linie de comandă, a secvenței următoare:* `sudo raspi-config`
2. *Navigarea prin listă se face cu tastele săgeata* ***sus/jos***
3. *Saltul de la o secțiune la alta se face cu tasta* ***Tab.***
4. *Confirmarea unei opțiuni selectate (supraluminată cu fond roșu) se face prin apăsarea tastei* ***Enter***

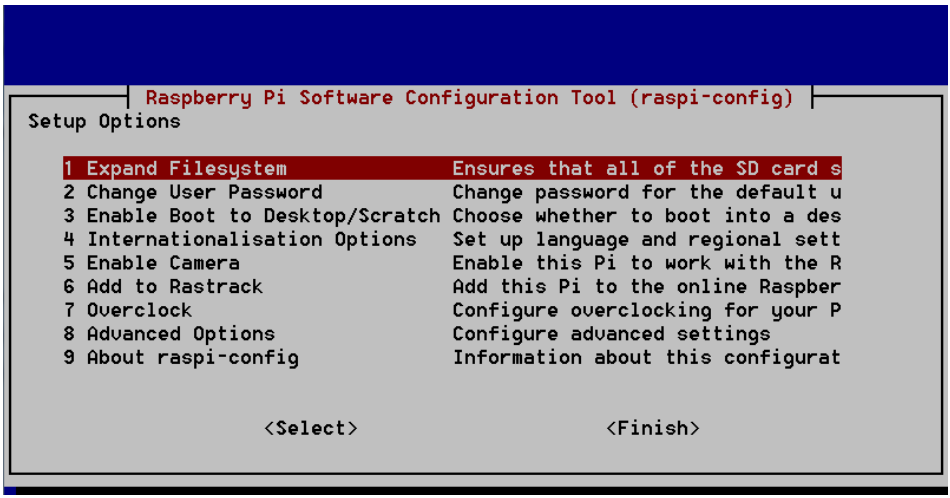


Fig. 6.14 Utilitarul de configurare raspi-config

Din cele nouă opțiuni afișate vom zăbovi asupra primelor trei.

6.5.1 **Expand Filesystem**

Dacă am folosit un card de memorie cu o capacitate mai mare de 4GB vom utiliza această opțiune pentru a extinde capacitatea sistemului de operare de a utiliza întregul spațiu de date disponibil.

6.5.2 **Change User Password**

Implicit sistemul se instalează cu un singur utilizator. Numele utilizatorului este **pi,** iar parola acestuia este **raspberry**. Dacă doriți să schimbați parola, aceasta este opțiunea de care aveți nevoie.

6.5.3 **Enable Boot to Desktop/Scratch**

Deși ultima în lista noastră, este cea mai importantă pentru scopul lucrării de față. Aceasta este opțiunea care ne oferă două posibilități nemaipomenite:

- Modul Desktop prin *Desktop Log in as user 'pi' at the graphical desktop* . Dacă ne dorim un calculator pe care să programăm în Scratch, să navigăm în Internet, să jucăm Minecraft sau să ne inițiem în tainele Linux, plus multe alte surprize, aceasta este opțiunea;
- Modul Scratch prin *Scratch Start the Scratch programming environment upon boot.* Dacă visăm la un calculator pe care să programăm numai și numai în Scratch, fără să ne distragă atenția nicio altă aplicație, această opțiune ne pune la dispoziție o jucărie care vorbește doar în limba pisoiului Scratch.

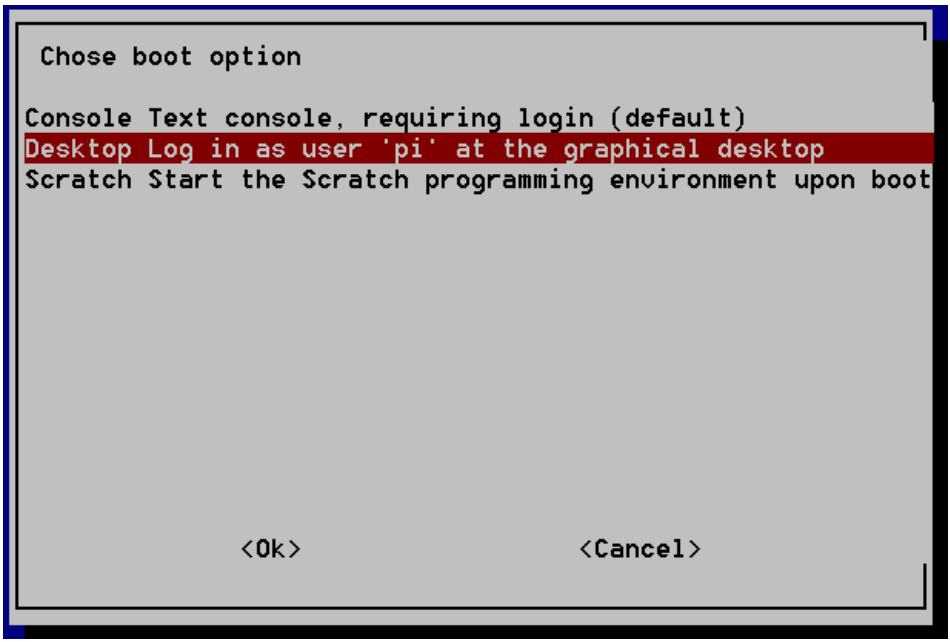


Fig. 6.15 Alegerea modului de start

***Observație**: Implicit, după instalare, opțiunea activă este pornirea în linie de comandă. Ca urmare este obligatoriu, dacă ne dorim Scratch, să comutăm pe una din celelalte două opțiuni.*

Ulterior dacă ne dorim să revenim la o altă opțiune, trebuie să știm cum accesăm linia de comandă:

- dacă suntem in mod Scratch la ieșirea din program prin combinația Ctrl+C, avem cinci secunde să comutăm în linie de comandă

- dacă suntem în mod Desktop accesăm pictograma de start a aplicației LxTerminal și, după pornirea aplicației, avem la dispoziție o linie de comandă în așteptare

După selectarea modului de lucru preferat, accesați butonul OK și confirmați restartarea sistemului.

Dacă selectați modul Desktop, după restart ecranul se va prezenta ca în imaginea de mai jos.

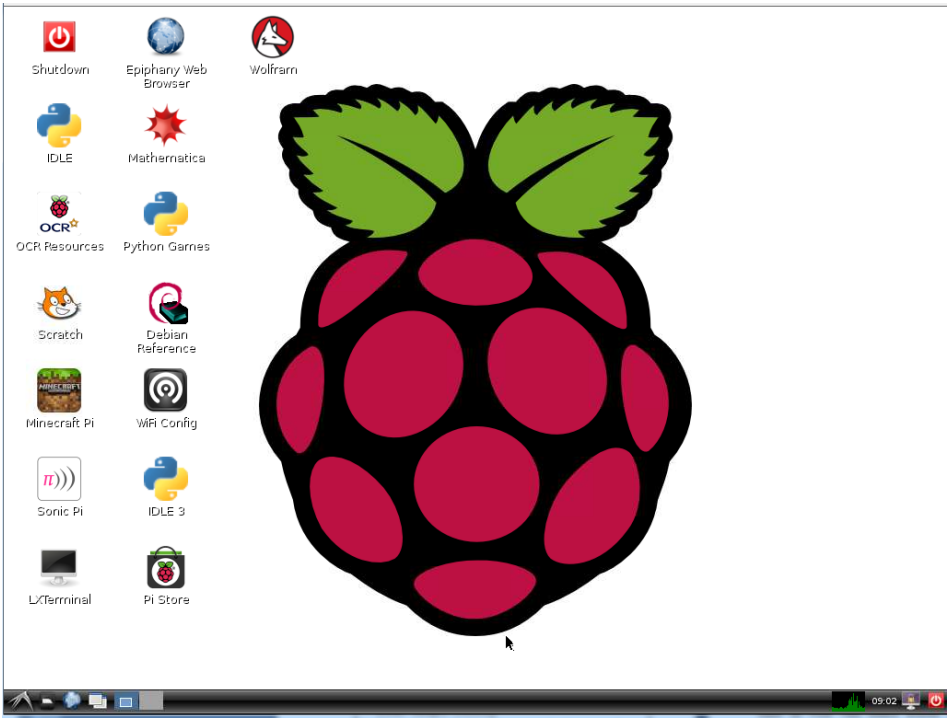


Fig. 6.16 Raspberry PI în mod Desktop

6.6 GPIO + Scratch

GPIO + Scratch a devenit realitate datorită efortului lui Simon Walters și s-a materializat în proiectul Scratch_GPIO. Blogul acestuia, http://cymplecy.wordpress.com/ este actualizat în mod constant cu informații privind evoluția proiectului. Găsiți aici idei noi, update-uri și răspunsuri la tot felul de probleme care pot fi rezolvate cu Scratch. Codul sursă pentru Scratch_GPIO este disponibil pe Github https://github.com/cymplecy/scratch_gpio .

6.6.1 **Avertizare**

Acest capitol este destinat cititorilor experimentați, cu cunoștinte de bază în domeniul electric. Pentru orice experiment, minorii vor trebui supravegheați permanent de un adult cu experiență în utilizarea și manipularea componentelor și echipamentelor electrice și electronice.

Deși spectaculos, controlul GPIO nu este lipsit de riscuri.

Autorii acestei lucrări nu își asumă în niciun fel răspunderea pentru consecințele determinate de conectarea și utilizarea greșită a unui circuit electric.

Înainte de orice acțiune, citiți și rețineți următoarele sfaturi.

ATENȚIE!

Utilizarea greșită a pinilor de pe headerul GPIO poate conduce la deteriorarea iremediabilă a Raspberry PI.

Toate acțiunile efectuate asupra pinilor GPIO trebuie realizate sub supravegherea unui adult.

Nu conectați/deconectați niciodată pini dacă Raspberry PI este alimentat cu curent.

Dacă ați introdus Raspberry PI într-un circuit electric prin intermediul pinilor GPIO, verificați cu atenție corectitudunea schemei înainte de a alimenta minicalculatorul.

În timpul utilizării evitați să atingeți cu elemente metalice pinii sau placa de baza. Se pot produce scurt circuite care vor deteriora echipamentul.

Pinii suportă o tensiune maximă de 3.3V. O tensiune mai mare va deteriora echipamentul.

Din fiecare pin se poate trage un curent de maxim 16mA. În plus, suma curenților pe toți pinii de ieșire nu poate depași 50mA.

6.6.2 Instalare ScratchGPIO5

Pentru a putea avea acces din Scratch la GPIO va trebui să utilizăm modul Desktop. În plus va trebui să instalăm aplicația ScratchGPIO5.

Observație: Dacă aveți cunoștințe medii de Linux, puteți configura accesul la GPIO și din modul Scratch. După instalarea ScratchGPIO5, trebuie doar să editați fișierul de boot /etc/profile.d/boottoscratch.sh pentru a rula ScratchGPIO5

6.6.2.1 Daca sunteți conectat la internet

Porniți LX Terminal și introduceți comenzile:

```
sudo wget http://goo.gl/Pthh62 -O isgh5.sh
sudo bash isgh5.sh
```

6.6.2.2 Daca NU sunteți conectat la internet

a) Inserați cardul într-un PC conectat la internet.
b) Descărcați fisierul **install_scratchgpio5.sh** de la adresa http://goo.gl/Pthh62
c) Salvați-l pe card
d) Extrageți cardul din PC și montați-l in Raspberry PI
e) Porniți LX Terminal și introduceți comanda

```
sudo bash /boot/install_scratchgpio5.sh
```

Aplicația de instalare va genera două noi pictograme pe Desktop:

1. ScratchGPIO5 – versiune pentru începători

2. ScratchGPIO5 Plus – versiune cu suport pentru module de extensie hardware; pentru avansați

Fig. 6.17 Pictogramele ScratchGPIO5

6.6.3 **Rularea unei aplicații Scratch la start**

Având ScratchGPIO5 instalat, dacă vă doriți ca o aplicație să ruleze la pornirea Raspberry PI, folosiți metoda următoare:

1. Porniți LX Terminal și introduceți comenzile:

```
mkdir -p /home/pi/.config/autostart
cd /home/pi/.config/autostart
touch scratchauto.sh
```

```
echo cd /home/pi/scratchgpio5 > scratchauto.sh
echo sudo python scratchgpio_handler5.py \& scratch \
"\""/home/pi/Documents/Scratch Projects/autostart.sb"\"" \
>> scratchauto.sh
chmod +x scratchauto.sh
```

2. Creați aplicația Scratch având grijă să includeți următoarele blocuri la începutul programului.

Salvați programul sub numele

/home/pi/Documents/Scratch Projects/autostart.sb

3. Restartați Raspberry PI. Programul pe care l-ați creat va rula automat.

6.6.4 Exemplul 1: Pisoiul miorlăitor și clipitor

Vom realiza un proiect care va permite aprinderea a două leduri la clic-stânga pe imaginea lui Scratch. Vom programa ca ieșiri pinii 11 și 13. Conectarea la masă se face prin pinul 6.

Materiale necesare:

- 3 fire de conexiune mamă-tată
- 2 rezistori de 270 Ohm
- 2 leduri
- un breadboard (dispozitiv ce permite conectarea fără lipituri a componentelor electronice)

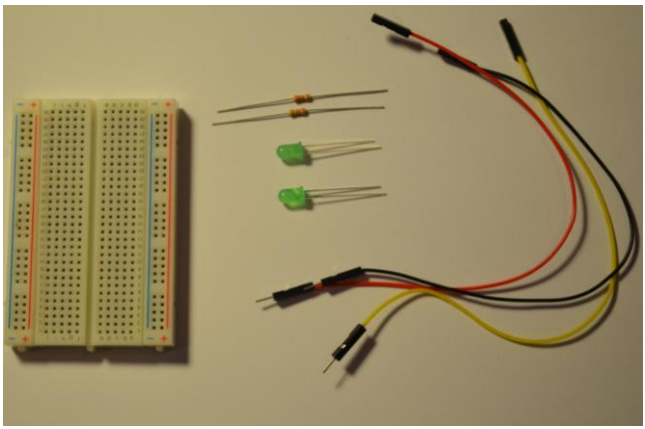


Fig. 6.18 Materialele necesare

Conectați componentele conform figurii de mai jos.

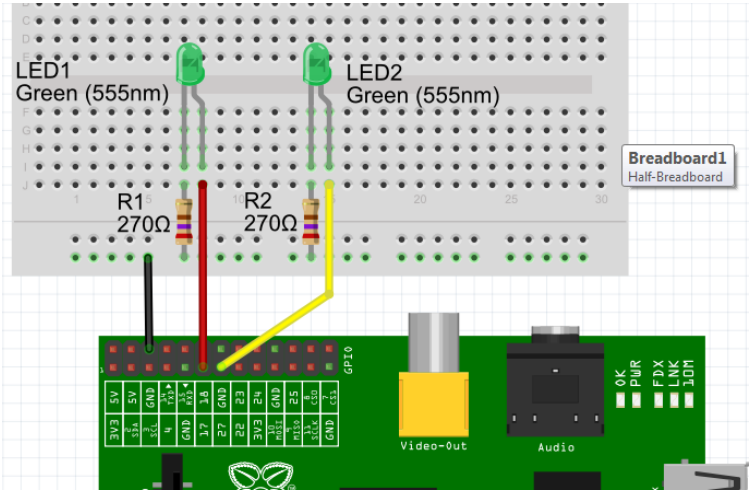


Fig. 6.19 Schema de conectare

Observație: Conectarea ledurilor se face întotdeauna cu pinul mai lung (ANOD) la plus, iar pinul mai scurt (CATOD) la minus. Dacă au o parte teșită, aceasta este în dreptul catodului.

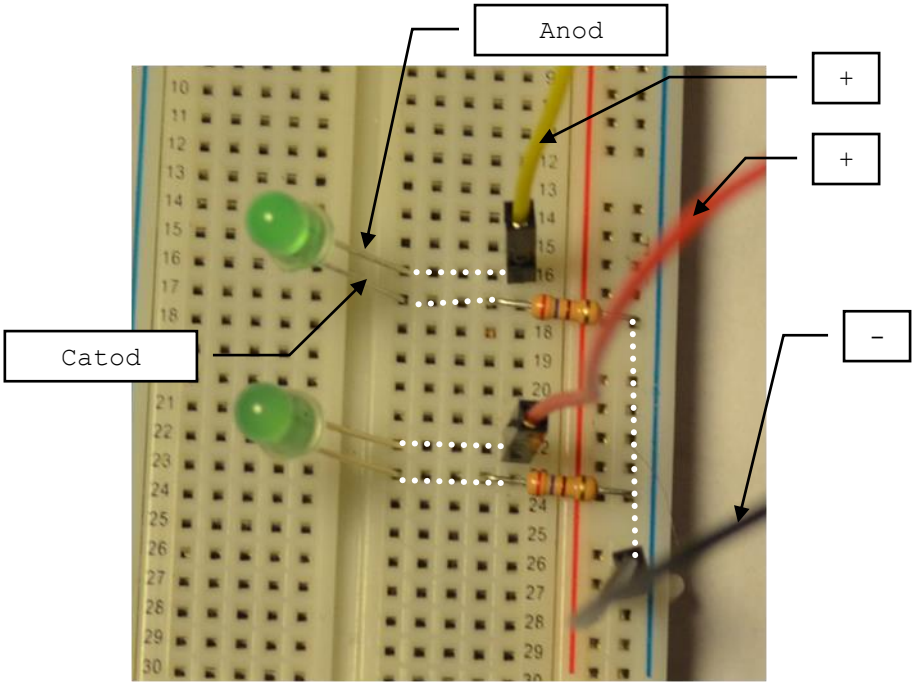


Fig. 6.20 Conectarea componentelor. Cu linie punctată este marcat traseul circuitului in breadboard

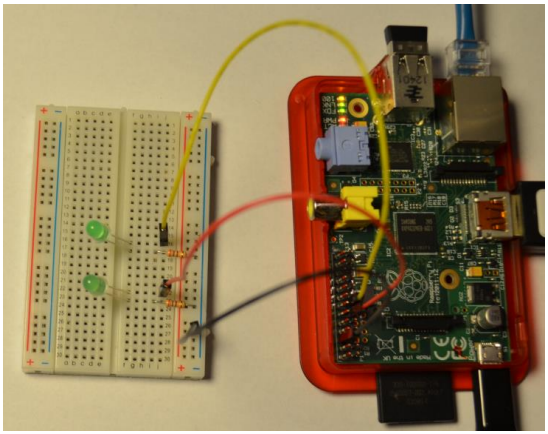


Fig. 6.21 Conectarea la Raspberry PI

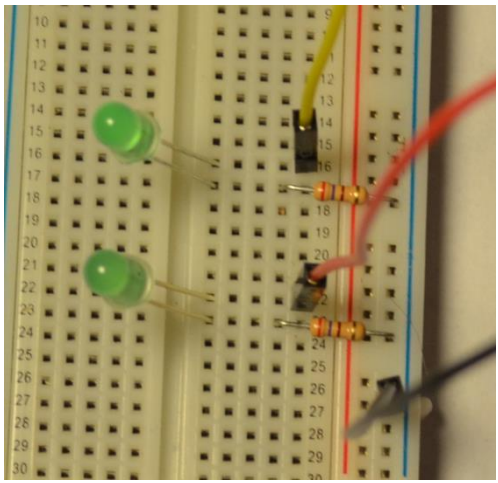


Fig. 6.22 Detaliu conectare in breadboard

După ce ați realizat conexiunile, porniți Raspberry PI. Porniți aplicația ScratchGPIO5.

Fig. 6.23 Pictograma ScratchGPIO 5

Creați un nou costum al pisoiului Scratch (costume3). Colorați ochii în verde conform figurii de mai jos.

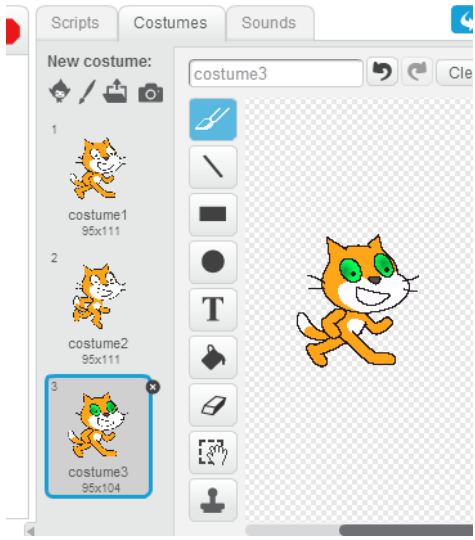


Fig. 6.24 Costume3 cu ochii colorați în verde

Realizați următorul script. Atenție la blocurile de broadcast care efectuează acțiuni cu pin11 și pin13

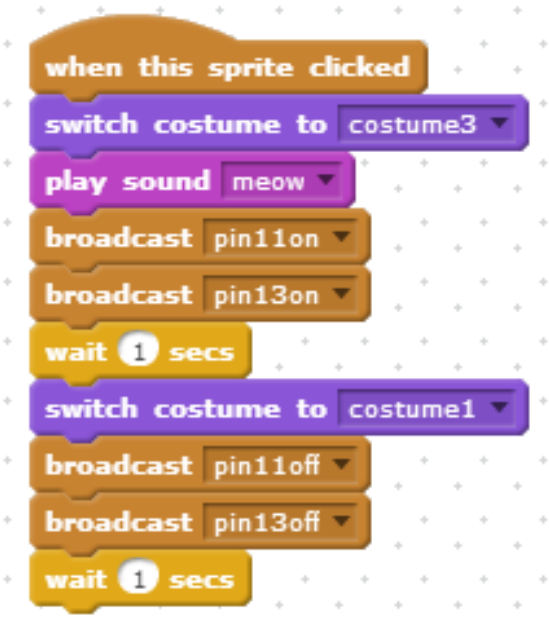


Fig. 6.25 Scriptul pentru comanda pinilor 11 și 13

Rezultatul aplicației: La orice atingere (mouse clic) a personajului Scratch, va rezulta un sunet “miau” iar ledurile se vor aprinde timp de o secundă.

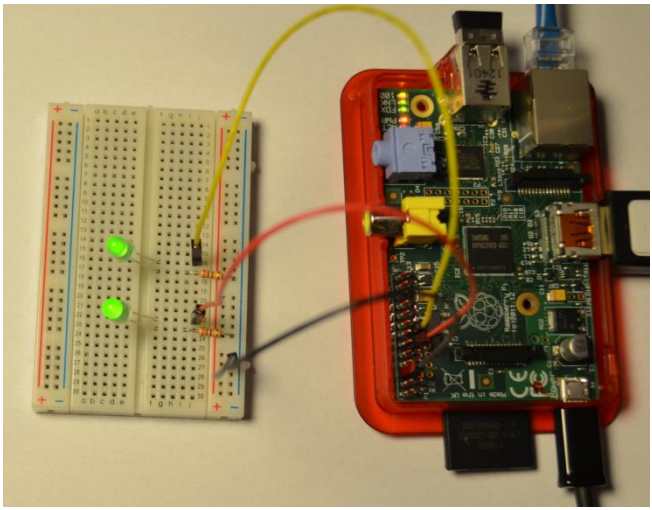


Fig. 6.26 Ledurile aprinse, comandate din Scratch

Cu un pic de imaginație, am putea să-l scoatem pe Scratch în afara calculatorului. Pe o bucată de plexiglas faceți două găuri cu un burghiu de 5mm (în lipsă puteți folosi carton și o foarfecă). Găurile să fie la o distanță aproximativ egală cu distanța dintre leduri. Desenați un cap de pisoi, ai cărui ochi să se potrivească cu cele două găuri.

Montați cele două leduri în găurile corespunzătoare.

Vom obține o pisică tocmai potrivită pentru o petrecere de Halloween.

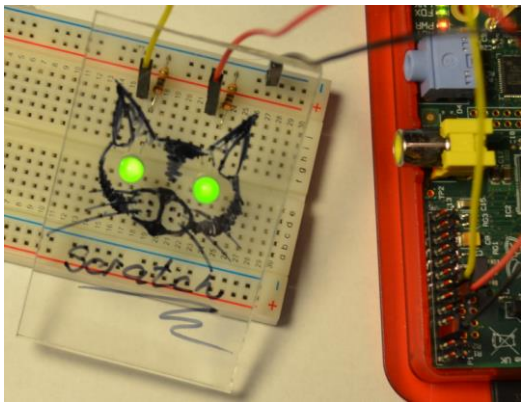


Fig. 6.27 Ledurile montate în placa de plexiglas

6.6.5 **Exemplul 2: Pisoiul cu telecomandă**

Această aplicație extinde exemplul 1 prin adăugarea unui buton de comandă. Apăsarea butonului va avea ca rezultat "aprinderea" ochilor atât pe imaginea lui Scratch din calculator, cât și pe ledurile din montaj.

Materiale necesare:

- 4 fire de conexiune mamă-tată
- 2 rezistori de 270 Ohm
- 1 rezistor de 1 KOhm
- 2 leduri
- 1 întrerupător
- un breadboard (dispozitiv ce permite conectarea fără lipituri a componentelor electronice)

Conectați componentele conform figurii de mai jos. Butonul de comandă se conectează la pinul 24.

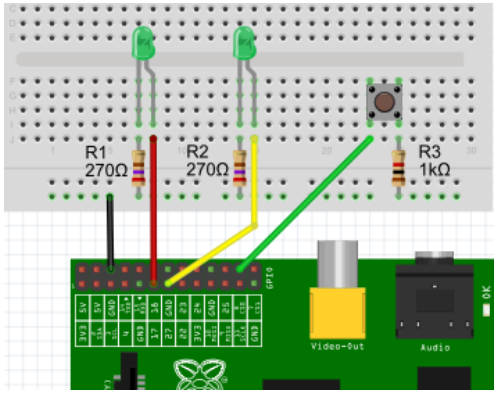
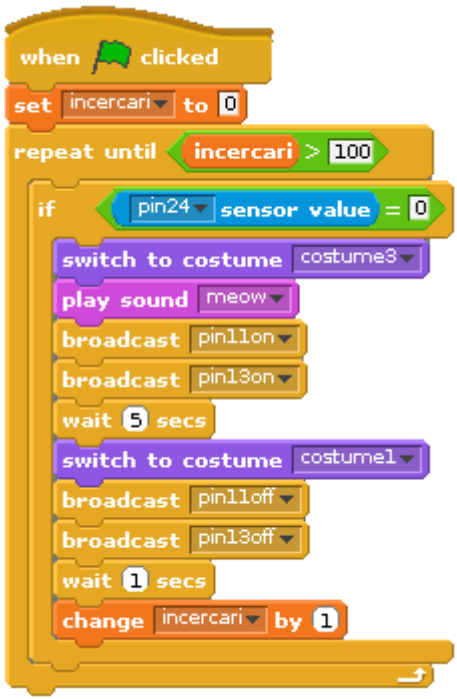


Fig. 6.28 Schema de conectare

Modificați scriptul de la exemplul 1 conform celor de mai jos:



6.29 Scriptul pentru controlul pinilor 11 și 13 prin evenimentele de pe pinul 24

Așa cum se poate vedea în script, aveți la dispoziție 101 încercari.

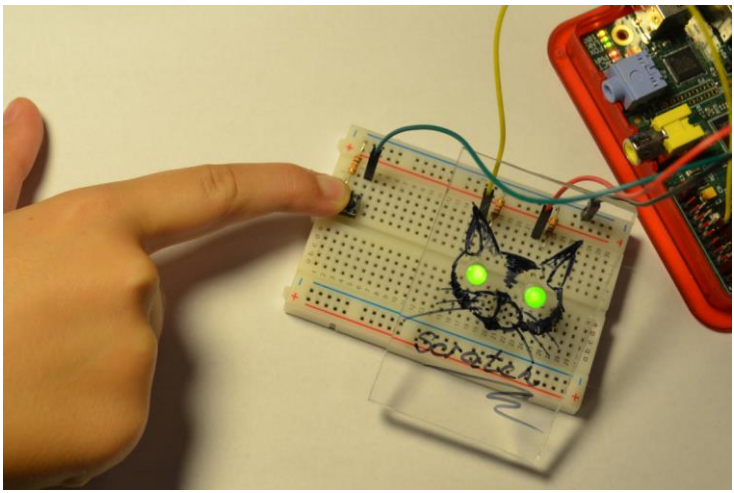


Fig. 6.30 Apăsarea butonului aprinde ledurile

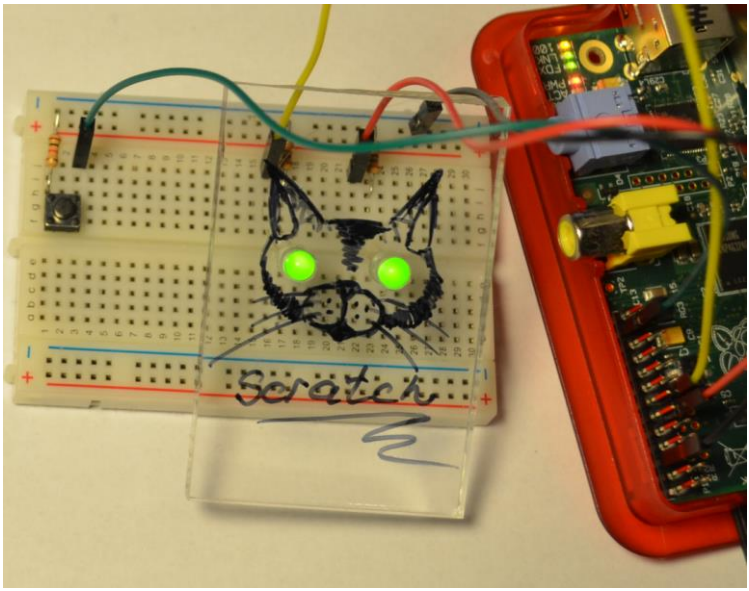


Fig. 6.31 Montajul finalizat

Capitolul 7. ***Sfaturi de la Mara…***

Suntem prieteni din anul 2012. A trecut ceva vreme și nu m-a dezamăgit niciodată. Am mare încredere în ea.

- ✓ Înainte de a începe să vă creați propriile personaje este bine să desenați mai întâi în Paint.
- ✓ Începeți cu programe simple. Apoi, încercați să le dezvoltați.
- ✓ Încercați să realizați propriile voastre jocuri. Satisfacție garantată!
- ✓ Povestiți prietenilor despre Scratch. Este mai distractivă munca în echipă!
- ✓ Nu uitați… Pentru orice programel este nevoie de imaginație și logică.

Capitolul 8. ***Bibliografie***

1. Jessica Chiang – Shall We Learn Scratch Programming
   http://shallwelearn.com/blog/stories/23343953-Shall-We-Learn-Scratch-Programming-eBook.pdf
2. **http://wiki.scratch.mit.edu/wiki/Scratch**
3. **http://info.scratch.mit.edu/Scratch2FAQ**
4. **http://info.scratch.mit.edu/Support/Scratch_Cards**
5. **http://ccdmures.ro/cmsmadesimple/pdf/jakab.pdf**
6. **http://ro.code.org**
7. **http://en.wikipedia.org/wiki/Raspberry_Pi**
8. **http://cymplecy.wordpress.com/scratchgpio/**