Activitats

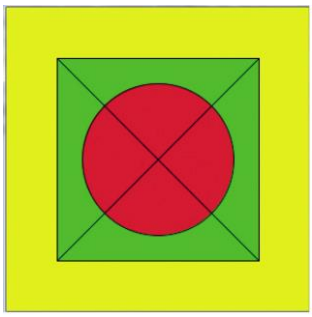
Practica i experimenta!
Et proposem que practiquis amb l'entorn integrat de desenvolupament (IDE) de Processing i que experimentis amb les diferents icones de la barra d'eines i algunes de les opcions dels menús existents. I la millor manera de fer-ho és creant el teu propi programa!

Pots començar escrivint i executant el programa d'exemple de la fig. 6.1 o un de semblant. Recorda que en el menú Ajuda (help) hi ha diverses opcions que et poden ajudar a resoldre dubtes o problemes.

Ara, fes algunes petites modificacions en el programa que acabes de crear per aconseguir un resultat similar al de la figura. Tanmateix, necessitaràs incorporar-hi aquestes dues noves ordres: background (red, green, blue) i fill (red, green, blue). Els paràmetres 'red', 'green' i 'blue' han de ser valors numèrics enters entre _ i 255. Investiga què fan aquestes ordres!

Programa de *Processing* per generar la figura:

```
background (255,255,0); //estableix el color groc com a fons
size(300,300); //mida de la finestra
fill(0,255,0); //color verd per omplir el quadrat
rect(50,50,200,200); //dibuixa un quadrat
fill(255,0,0); //color vermell per omplir el cercle
ellipse(150,150,150,150); //dibuixa un cercle
line(50,50,250,250); //dibuixa una línia obliqua
line(50,250,250,50); //dibuixa una línia obliqua
```



1. **Cerca informació sobre les característiques dels llenguatges de programació interpretats i compilats i compara'n els principals avantatges i inconvenients. Quins solen ser més ràpids?**

Resposta oberta.

Llenguatges interpretats. Un programa auxiliar (intèrpret) executa les instruccions del programa de manera directa. Exemples: BASIC, Lisp, PHP, Perl, Python…

Llenguatges compilats. Un programa annex, anomenat compilador, tradueix el codi font en codi màquina. L'arxiu resultant es pot executar directament. Exemples: C, Pascal, Cobol...

Els llenguatges compilats són més fàcils de depurar, es pot fer un procés de depuració pas a pas, consultant valors de variables, ... en un interpretat això no és tan fàcil.

En els compilats és possible ajuntar codi de diferents arxius, llibreries, classes, ... en els interpretats, també és més difícil.

El procés de creació del programa, suposant que no cal fer una gran depuració d'errors és més ràpid en l'interpretat, però l'execució del programa és més ràpida en el compilat;

© Mcgraw-Hill Education

entre altres coses perquè a la compilació hi ha una fase d'optimització de trucades a subrutines i d'optimització, en general, del codi.

Solen ser **més ràpids els llenguatges compilats**.

Practica i experimenta

Fes el programa que et proposem a continuació per practicar l'ús de variables del sistema i veure'n el resultat:

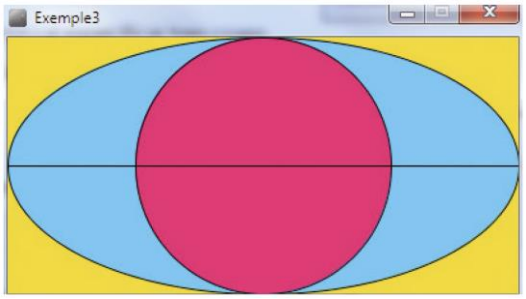
```
size(400, 200);
ellipse(width/2, height/2, width, height);
```

Ara, canvia la mida de la finestra en píxels: size(amplada, altura) i comprova'n el nou resultat. Què ha canviat?

Investiga com funciona l'ordre ellipse(x, y, x diàmetre, y diàmetre) quan es modifiquen els paràmetres.

Per acabar, fes una nova versió de l'anterior programa per obtenir una solució similar a la que mostra la figura. Utilitza les variables del sistema with i height i procura que, sigui quina sigui la mida de la finestra, el resultat final sigui semblant.

Programa de *Processing* per generar la figura:




```
size(400,200);
int variable1 = 0;
int variable2 = 100;
background(255, 204, 0);
fill(0,200,255);
ellipse(width/2, height/2, width, height);
fill(255,0,100);
ellipse(width/2, height/2, width/2, height);
line(0, height/2, width, height/2);
```

Practica i experimenta

Fes el programa que et proposem, a continuació, per calcular la longitud i l'àrea d'una circumferència i experimenta què succeeix:

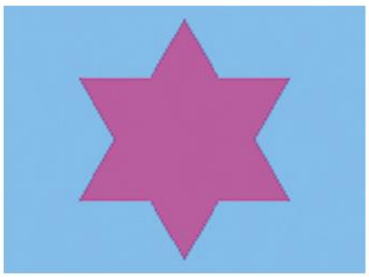
```
float radi=10, area, longitud;   // assignació valor radi i declar. variables
longitud = 2 * PI * radi;   // càlcul de la longitud de la circumferència
println("Longitud = " + longitud);   // escriu el resultat de la longitud
area = PI * radi * radi;   // càlcul de l'àrea
println("Àrea = " + area);   // escriu el resultat de l'àrea
```

© Mcgraw-Hill Education

Ara, crea un nou programa per calcular l'àrea d'un triangle i d'un rectangle. Sabries dibuixar una estrella com la de la figura? Intenta-ho!

Programa de *Processing* per generar la figura:

```
size(180,180);
background(0,190,255);
noStroke();
fill(255,0,240);
triangle(90,27,40,113,140,113);
triangle(40,55,140,55,90,141);
```



2. **Realitza un programa que donades tres variables: A=4, B=7 i C=0, intercanviï els valors de A i B i assigni finalment a C el valor del triple de la suma d'A més B. Escriu per pantalla els valors finals de les tres variables.**

Programa amb *Processing*:

```
int A=4, B=7, C=0;
C=A;
A=B;
B=C;
C=(A+B)*3;
println("A =",A);
println("B =",B);
println("C =",C);
```

3. **Digues, per a cada cas, si el codi de programa és correcte o no. En cas afirmatiu, explica què mostrarà per pantalla i, en cas negatiu, explica per què no funciona:**

a) **int A = 3.14; println(A);** → *No és correcte!* La variable "A" es numèrica de tipus enter (int) i no se li pot assignar un nombre decimal: 3.14
b) **boolean resultat = (1 > 4); println(resultat)** → *Correcte!* Mostrarà: *False*
c) **char paraula = «hola»; println(paraula)** → *No és correcte!* La variable "paraula" es de tipus caràcter (char) i no se li pot assignar la paraula "hola".

4. **Analitza el programa adjunt i explica què fa. Comprova'n el funcionament.**

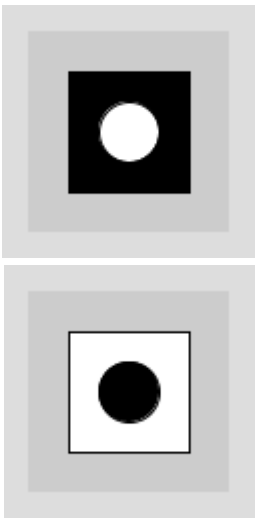
```
int valor = 0;
void draw() {
fill(valor);
rect(20, 20, 60, 70); }
void keyPressed() {
if (valor == 0) {
valor = 255;
} else {
valor = 0; } }
```

© Mcgraw-Hill Education

Cada cop que es prem una tecla el rectangle dibuixat a la finestra canvia de color (de negre a blanc i viceversa). Inicialment, el rectangle és de color negre; en prémer una tecla canvia a color blanc... i així successivament.

Com el modificaries perquè hi dibuixés dues figures geomètriques i intercanviés el color d'aquestes cada cop que es premés una tecla?

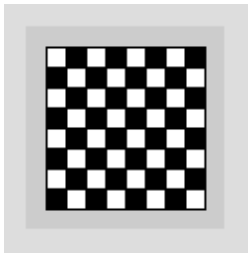
```
int valor1 = 0, valor2 = 255;
void draw() {
fill(valor1);
rect(20,20,60,60);
fill(valor2);
ellipse(50,50,30,30); }

void keyPressed() {
if (valor1 == 0) {
valor1 = 255;
valor2 = 0;
} else {
valor1 = 0;
valor2 = 255; } }
```



5. **Examina el funcionament en Processing dels dos exemples anteriors d'estructura «for» i d'estructura «while».**

Activitat pràctica i d'aprenentatge procedimental.

6. **Fes servir l'editor de l'IDE de Processing per escriure i executar un programa que dibuixi les 64 caselles (8 x 8) d'un tauler d'escacs utilitzant l'estructura de repetició «for» o «while». Sabries fer que també dibuixés el fons de les caselles (blanques i negres)?**



```
//Dibuixa les 64 caselles del tauler d'escacs
size(100,100); //mida de la pantalla
int color1=255, color2=0, color3=0;
for (int i = 1; i < 20; i = i+1) {
  line(10,i*10,90,i*10);
}
for (int i = 1; i < 20; i = i+1) {
  line(i*10,10,i*10,90);
}
//Dibuixa el fons de les caselles (blanc i negre)
for (int j = 1; j < 9; j = j+1) {
```

© Mcgraw-Hill Education

```
for (int i = 1; i < 9; i = i+2) {
  fill(color1);
  rect(i*10,j*10,10,10);
  fill(color2);
  rect((i+1)*10,j*10,10,10);
}
  color3=color1; color1=color2; color2=color3;
}
```

7. **Verifica el funcionament d'aquest programa i de la funció associada «dibuixaMascota», amb paràmetres i sense retorn de valors, per dibuixar una mascota com la de la figura en un lloc determinat de la pantalla.**

Activitat pràctica de verificació del funcionament d'un programa i de la funció associada «dibuixaMascota», amb paràmetres i sense retorn de valors, per dibuixar una mascota com la de la figura en un lloc determinat de la pantalla.

```
void setup() {   size(400,400);   } //mida de la pantalla en píxels: 400 x 400
void draw() { //funció principal del programa que s'executa contínuament
  background(0); //estableix el color de fons de la finestra de dibuix
  dibuixaMascota(width/2,height/2);   } //dibuixa la mascota en el centre
void dibuixaMascota(float x, float y) { //funció per dibuixar mascota
  strokeWeight(3); //amplada del traç de dibuix
  stroke(255); //color del traç per dibuixar línies i formes (blanc)
  fill(255,0,0); //color per omplir les formes geomètriques (vermell)
  line(x-30,y-30,x+25,y+25); //dibuixa una línia al cap de la mascota
  line(x-25,y+25,x+30,y-30); //dibuixa una línia al cap de la mascota
  ellipse(x,y,50,50); //dibuixa una circumferència (cap mascota)
  point(x-10,y-5); //dibuixa un punt (ull esquerre)
  point(x+10,y-5); //dibuixa un punt (ull dret)
  line(x-7,y+10,x+7,y+10);   } //dibuixa una línia (boca)
```

8. **Crea la funció *dibuixaEmoticona()*, sense paràmetres, que dibuixi, al centre de la pantalla, un Pac-Man o «menjacocos» semblant al de la figura. O bé, si ho prefereixes, pots dissenyar la teva pròpia emoticona.**

```
void dibuixaEmoticona() {
  size(400,400); //mida de la pantalla
  background(255); //color blanc fons pantalla
  strokeWeight(3); //amplada del traç de dibuix a 3
  stroke(0); //color del traç per dibuixar línies i formes
  fill(255,255,0); //color per omplir les formes
  ellipse(200,200,80,80); //dibuixa una circumferència
```

© Mcgraw-Hill Education

```
  fill(255,255,255); //color per omplir les formes
  noStroke();  //sense línia de traç
  triangle(200,200,200+50,200-30,200+50,200+30);  //dibuixa un triangle
  stroke(0); //color del traç per dibuixar línies i formes
  line(200,200,200+33,200-21); //dibuixa una línia
  line(200,200,200+33,200+21); //dibuixa una línia
  strokeWeight(10); //amplada del traç de dibuix a 10
  point(200,200-16); //dibuixa un punt (ull dret)
}
```

9. **Sabries crear un bloc de programa que fes que l'emoticona que has creat amb la funció *dibuixaEmoticona()* es desplacés horitzontalment per la pantalla com en un videojoc?**

Primer caldria modificar la funció de l'activitat anterior i incorpora-li dos paràmetres (x, y) perquè dibuixi l'emoticona en la part de la pantalla fixada per les coordenades «x, y». Ara, la funció l'haurem de cridar així: *dibuixaEmoticona(a, b),* on «a» i «b» són els valors que s'assignaran als paràmetres «x, y» de la funció i que determinaran la posició de l'emoticona a la pantalla.

```
void dibuixaEmoticona(float x, float y) {
  size(400,400); //mida de la pantalla
  background(255); //color blanc fons pantalla
  strokeWeight(3); //amplada del traç de dibuix a 3
  stroke(0); //color del traç per dibuixar línies i formes
  fill(255,255,0); //color per omplir les formes
  ellipse(x,y,80,80); //dibuixa una circumferència (cap)
  fill(255,255,255); //color per omplir les formes
  noStroke();  //sense línia de traç
  triangle(x,y,x+50,y-30,x+50,y+30);  //dibuixa un triangle
  stroke(0); //color del traç per dibuixar línies i formes
  line(x,y,x+33,y-21); //dibuixa una línia
  line(x,y,x+33,y+21); //dibuixa una línia
  strokeWeight(10); //amplada del traç de dibuix a 10
  point(x,y-16); //dibuixa un punt (ull dret)
}
```

A continuació, s'hauria d'afegir un bloc de programa que fes que l'emoticona que s'ha creat amb la funció *dibuixaEmoticona(x,y)* es desplacés horitzontalment per la pantalla com en un videojoc.

```
int pos, vel; //definició de les variables 'pos' i 'vel'
void setup(){
```

© Mcgraw-Hill Education

```
  size(800,400); //mida de la pantalla
  pos = 0; vel = 5;  //incialitzar variables posició i velocitat
}
void draw(){
  background(255); //color del fons de la pantalla
  pos = pos + vel; //increment de la posició 'x'
  dibuixaEmoticona(pos, height/2);
  if((pos>width)){pos=0;} //condició per tornar a l'inici
}
void dibuixaEmoticona(float x, float y) {
  strokeWeight(3); //amplada del traç de dibuix a 3
  stroke(0); //color del traç per dibuixar línies i formes
  fill(255,255,0); //color per omplir les formes
  ellipse(x,y,80,80); //dibuixa una circumferència (cap)
  fill(255,255,255); //color per omplir les formes
  noStroke();  //sense línia de traç
  triangle(x,y,x+50,y-30,x+50,y+30);  //dibuixa un triangle
  stroke(0); //color del traç per dibuixar línies i formes
  line(x,y,x+33,y-21); //dibuixa una línia
  line(x,y,x+33,y+21); //dibuixa una línia
  strokeWeight(10); //amplada del traç de dibuix a 10
  point(x,y-16); //dibuixa un punt (ull dret)
}
```

10. **Comprova el funcionament del programa que es descriu a continuació, el qual inclou una funció per calcular el percentatge d'una quantitat**

Activitat pràctica de comprovació del funcionament del programa que es descriu a continuació, el qual inclou una funció per calcular el percentatge d'una quantitat.

```
float valor; float quant; float percent; //declaració de variables
void setup() { //declaració de variables
  quant = 500; //declaració de variables
  percent = 21; //declaració de variables
  valor = calculPercentatge(quant,percent); //declaració de variables
  println(valor); } //declaració de variables
float calculPercentatge(float quantitat, float tantxcent) { //declaració de variables
  float calcul; //Posa el color negre
  calcul = quantitat * tantxcent / 100; //Posa el color negre
  return calcul;   } //retorna el valor calculat per la funció
```

© Mcgraw-Hill Education

11. **Crea la funció calculaMitjana(a,b) que rebi dos paràmetres (a i b) i que retorni com a resultat la mitjana aritmètica d'aquests dos valors.**

```
float valor; float a; float b;
void setup() {
  a = 224;
  b = 116;
  valor = calculaMitjana(a,b);
  println(valor);
}
float calculaMitjana(float a1, float b1) {
  float calcul;
  calcul = (a1 + b1) / 2;
  return calcul;
}
```

© Mcgraw-Hill Education

Supera el repte!

1. Crea, practica i experimenta!

Et proposem el repte de dissenyar un programa de joc d'ordinador.
La versió bàsica del joc suggerit consistirà a intentar «caçar» emoticones (amb un estri que només es pot desplaçar horitzontalment per la part inferior de la pantalla) que aniran caient verticalment i amb trajectòries aleatòries des de dalt. L'emoticona serà «caçada» si aconseguim tocar-la amb l'estri. Pots fer totes les variacions que consideris adients... o inventar-te un nou joc. Aquests són alguns consells que et poden ajudar en la seva elaboració:
a) Primer, crea la funció dibuixaEmoticona(x,y) per dibuixar una emoticona, una mascota o qualsevol altra imatge de creació pròpia. La funció contindrà dos paràmetres (x, y) per dibuixar la imatge a la posició determinada per les coordenades «x, y».

```
void dibuixaEmoticona(float x, float y){
  noStroke();
  fill(255,204,0); //color groc per omplir les formes
  ellipse(x,y,80,80); //dibuixa una circumferència (cap)
  fill(0); //color negre per omplir les formes
  ellipse(x-15,y-10,10,20); //dibuixa ull esquerra
  ellipse(x+15,y-10,10,20); //dibuixa ull dret
  noFill(); stroke(0); strokeWeight(3);
  arc(x,y,55,55, QUARTER_PI-0.4,PI-QUARTER_PI+0.4); //dibuixa boca
}
```

b) Dissenya una altra funció dibuixaEstri(mx,my) per dibuixar l'estri per «caçar» emoticones, el qual serà representat a la posició de la pantalla determinada pels valors dels paràmetres 'mx' i 'my'.

```
void dibuixaEstri(float mx, float my){
  stroke(255);
  fill(205,0,0); //color vermell per omplir les formes
  triangle(mx, my, mx-30,my+30,mx+30,my+30); //dibuixa un triangle
}
```

c) Decideix i declara totes les variables del programa. Per exemple:

```
int posX=400, posY=-30, velY=5, punts=0, totEmot=10, numEmot=0;
```

d) Especifica les principals estructures de control que hauràs de fer servir.

```
if(numEmot<totEmot) {
  background(0); //color negre de fons de pantalla
  dibuixaEmoticona(posX, posY);
  dibuixaEstri(mouseX, height-50);

  if((posX >= mouseX-15) && (posX<=mouseX+15) && (posY >= height-40)) {
    posX=round(random(width)); posY=-40; velY=velY+1; punts=punts+1;
```

© Mcgraw-Hill Education

```
    numEmot=numEmot+1;
  }

  if(posY>(height+40)){
  posY = -40; posX=round(random(width)); numEmot=numEmot+1;
  }

  posY = posY + velY;
  text("Punts: "+punts,width-120, 30);
  text(numEmot+" / "+totEmot, width-100, height-20);

}  else {
    text("F I N A L  D E L  J O C !", width/2-150, height/2);
    }
```

e) Programa final, un cop integrats els diferents mòduls i depurat el programa:

```
int posX=400, posY=-30, velY=5, punts=0, totEmot=10, numEmot=0;

void setup(){
  size(800,600); //mida de la pantalla
  textSize(25);
  smooth();
}

void draw() {
  if(numEmot<totEmot) {
  background(0); //color negre de fons de pantalla
  dibuixaEmoticona(posX, posY);
  dibuixaEstri(mouseX, height-50);
  if((posX >= mouseX-15) && (posX<=mouseX+15) && (posY >= height-40)) {
    posX=round(random(width)); posY=-40; velY=velY+1; punts=punts+1;
    numEmot=numEmot+1;
  }
  if(posY>(height+40)){
  posY = -40; posX=round(random(width)); numEmot=numEmot+1;
  }
  posY = posY + velY;
  text("Punts: "+punts,width-120, 30);
  text(numEmot+" / "+totEmot, width-100, height-20);
  } else {
    text("F I N A L  D E L  J O C !", width/2-150, height/2);
    }
}

void dibuixaEmoticona(float x, float y){
  noStroke();
  fill(255,204,0); //color groc per omplir les formes
  ellipse(x,y,80,80); //dibuixa una circumferència (cap)
  fill(0); //color negre per omplir les formes
  ellipse(x-15,y-10,10,20); //dibuixa ull esquerre
  ellipse(x+15,y-10,10,20); //dibuixa ull dret
```

© Mcgraw-Hill Education

```
  noFill(); stroke(0); strokeWeight(3);
  arc(x,y,55,55, QUARTER_PI-0.4,PI-QUARTER_PI+0.4);//dibuixa boca
}

void dibuixaEstri(float mx, float my){
  stroke(255);
  fill(205,0,0); //color vermell per omplir les formes
  triangle(mx, my, mx-30,my+30,mx+30,my+30); //dibuixa un triangle
}
```

© Mcgraw-Hill Education

Examina't!

E 1. **Al conjunt d'instruccions escrites per realitzar una tasca específica en un ordinador, en un telèfon o en altres aparells tecnològics programables l'anomenem:**

a) Llenguatge de programació.
b) Programa informàtic.
c) Estructura de control.
d) Llenguatge màquina.

Resposta correcta: b) Programa informàtic.

E 2. **Comenta breument quina és la funció de les dades i per què són importants en la programació d'aplicacions. Posa alguns exemples de tipus de dades que fan servir els llenguatges de programació.**

Els programes i les aplicacions dels ordinadors i dispositius mòbils fan servir gran quantitat de dades, que són símbols (numèrics, alfabètics, etc.) que descriuen fets, condicions, situacions o valors de manera adequada per poder ser tractades i processades per aquests dispositius digitals. Un conjunt de dades significatives que resulten útils es converteixen en informació. Una dada elemental és la mínima informació que es té en un programa.

Exemples de tipus de dades:

- **int**: representa un valor numèric enter, positiu o negatiu, sense cap decimal.
- **float**: representa un valor numèric real, positiu o negatiu, amb decimals.
- **char**: representa un caràcter: lletra de l'alfabet, dígit, signe...
- **boolean**: representa un valor booleà o de tipus lògic. Pot ser cert (true) o fals (false).
- **String**: representa una cadena o seqüència de caràcters.

E 3. **Escriu dos o tres exemples d'ús per a cada tipus d'operador i especifica l'operació que realitzen:**

Tipus operador	Operador	Exemple d'ús	Operació
Aritmètic	**+**	A + B	Suma
Aritmètic	**-**	A – B	Resta
Aritmètic	*****	A * B	Multiplicació
Aritmètic	**/**	A / B	Divisió
Relacional	**>**	A > B	A major que B
Relacional	**<**	A < B	A menor que B
Relacional	**!=**	A != B	A distint de B
Relacional	**==**	A == B	A igual que B

© Mcgraw-Hill Education

Lògic	**&&**	A && B	Compara dos operands o expressions. Retorna *true* si tots dos són *true*, i *false* en cas contrari.
Lògic	**||**	A || B	Compara dos operands o expressions. Retorna *true* si un dels dos és *true*, i *false* si ambdós són *false*.
Lògic	**!**	! A	Inverteix el valor booleà de l'operand o expressió. Retorna *true* si l'operand és *false*, i a l'inrevés.

E 4. **Quin serà el valor de les variables booleanes *varbool1*, *varbool2* i *varbool3* un cop executades les instruccions següents?**

```
int var1 = 5,  var2 = 10;
boolean varbool1, varbool2, varbool3;
varbool1 = var2 < var1;
varbool2 = var1 != var2;
varbool3 = (var2 / 2 == var1);
```

varbool1 = **false**
varbool2 = **true**
varbool3 = **true**

E 5. **Examina el programa següent i explica què fa.**

```
float mida = 1;
void setup(){ size(400,400); }
void draw(){
  background(0);
  stroke(255);
  ellipse(width/2, height/2, mida, mida);
  if(mida < 350) { mida = mida + 0.5; }
    else { mida = 1; }
}
```

El programa simula l'inflament d'un globus cilíndric de color blanc sobre un fons negre, és a dir, apareix per pantalla una rodona blanca petita que cada cop es va fent més gran. Quan arriba a la mida màxima, la seqüència torna a començar... i així successivament fins que aturem el programa.

© Mcgraw-Hill Education

E 6. **Explica quina és la diferència entre una variable i una constant.**

Una *variable* és una dada o una estructura de dades emmagatzemada a la memòria de l'ordinador que pot veure modificat el seu valor en qualsevol moment durant l'execució del programa. En canvi, una *constant* és un tipus especial de variable el valor de la qual només pot ser llegit, però mai modificat. S'utilitza en dades que mai varien: IVA, PI, etc.

E 7. **Dissenya i escriu les instruccions que formaran la funció calculaAreaTriangle(a, b), la qual rebrà dos paràmetres 'a' (altura d'un triangle) i 'b' (base d'un triangle) i retornarà com a resultat l'àrea d'aquesta figura geomètrica.**

```
float altura=10, base=5, resultat=0;
void setup() {
  resultat = CalculaAreaTriangle(altura,base);
  println(resultat);
}

float CalculaAreaTriangle(float a, float b) {
  float area;
  area = b * a / 2;
  return area;
}
```

E 8. **Corregeix els sis errors d'aquest programa perquè, cada cop que s'executi, generi una finestra amb una imatge artística aleatòria semblant a la de l'exemple de la fig. 6.10.**

```
size(300,300);
background(255);
for(int i = 0; i < 60; i = i + 1)  {
  noStroke()
  fill(random(255), random(255), random(255));
  ellipse(random(height), random(width), i, r);
}
```

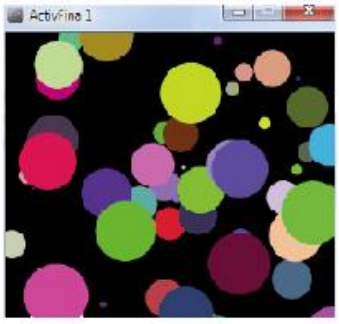


Fig. 6.10.

De fet només n'hi ha 4 errors, com a tals, en el programa. Els altres dos suposats "errors" són dues diferències amb el programa original: la mida de la finestra (200x200) i el nombre de cercles de colors (40) que hi apareixen:

© Mcgraw-Hill Education

```
size(200,200);
background(0);
for(int i = 0; i < 40; i = i + 1) {
noStroke();
fill(random(255), random(255), random(255));
ellipse(random(height), random(width), i, i);
}
```

E 9. **Desenvolupa un senzill programa perquè, cada cop que s'executi, dibuixi aleatòriament deu boles de colors diversos, de diferent mida i en posicions diferents. I, a continuació, que escrigui per pantalla cinc nombres (de l'1 al 5) en posicions aleatòries.**

Pots utilitzar aquest programa com un joc en què poden participar diversos jugadors. Cada jugador executa un cop el programa (o un nombre determinat de vegades que heu d'establir al principi). A cada jugada, se sumen els punts dels nombres que toquen totalment o parcialment alguna bola de color. En l'exemple de la fig. 6.11, només el nombre '1' toca alguna bola de color. Per tant, el nombre total de punts obtinguts en aquesta jugada ha estat de només un. La màxima puntuació en una jugada podrà ser, per tant, de 15 punts (1+2+3+4+5=15).

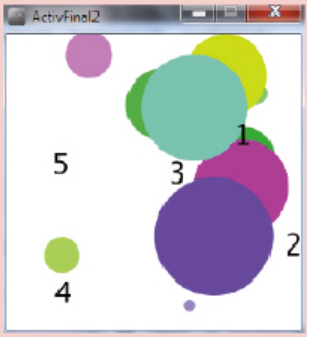


Fig. 6.11.

```
size(250,250);
background(255);
textSize(25);
for(int i=1; i<11; i=i+1) {
  noStroke();
  fill(random(255), random(255), random(255));
  ellipse(random(width), random(height), i*10, i*10);
}
for(int i=1; i<6; i=i+1) {
  stroke(0);
  fill(0);
  text(i,random(width),random(height));
}
```

© Mcgraw-Hill Education