

# Programação Multimédia

## Módulos

# Conceitos Básicos de Um Programa

- Memória (armazenamento de dados)
  - Guardar dados temporários (durante a execução do programa)
- Selecção (...de caminhos de execução)
  - O nosso programa pode ter ramos que são executados em determinadas circunstâncias
- Iteracção (execução repetida das instruções)
  - Executar várias vezes a mesma acção (sobre dados diferentes)
- **Módulos**
  - **Reutilizar conjuntos de instruções (com parametrização)**

# Exemplo #1

```
void setup() {
    size(400, 400);
}

void draw() {
    background(0);

    // desenhar uma 'cara'
    fill(255, 255, 255);    //branco
    rect(0, 0, 50, 50);    //face

    fill(0, 255, 0);       //verde
    rect(10, 10, 10, 10);  //olho esquerdo
    rect(30, 10, 10, 10); //olho direito

    fill(255, 0, 0);       //vermelho
    rect(15, 30, 20, 10); // boca
}
```

# Exercício #1

- Modificar o programa anterior para desenhar 3 caras em posições diferentes

# Problema

- E se, agora, quiséssemos uma cara oval?
- E se pretendéssemos desenhar 10 caras?

# Módulos

- É frequente repetirmos código que faz a mesma operação
  - Possivelmente sobre dados diferentes
- Em vez de escrevermos repetidamente o mesmo código podemos dar um nome a uma secção de código e invocar essa secção

# Métodos

- Em Processing, os módulos, chamam-se **métodos**
- Outros nomes para módulos
  - Funções
  - Procedimentos

# Exemplo #2

```
void setup() {
  size(400, 400);
}

void draw() {
  background(0);

  desenhaCara();
}

void desenhaCara() {
  // desenhar uma 'cara'
  fill(255, 255, 255);    //branco
  rect(0, 0, 50, 50);    //face

  fill(0, 255, 0);       //verde
  rect(10, 10, 10, 10);  //olho esquerdo
  rect(30, 10, 10, 10);  //olho direito

  fill(255, 0, 0);       //vermelho
  rect(15, 30, 20, 10);  // boca
}
```

# Exemplo #3

```
void setup() {
    size(400, 400);
}

void draw() {
    background(0);

    desenhaCara(0, 0);

    desenhaCara(100,

    desenhaCara(200,

void desenhaCara(int x, int y) {
    // desenhar uma 'cara'
    fill(255, 255, 255);      //branco
    rect(x, y, 50, 50);      //face

    fill(0, 255, 0);         //verde
    rect(x+10, y+10, 10, 10); //olho
    rect(x+30, y+10, 10, 10); //olho direito

    fill(255, 0, 0);         //vermelho
    rect(x+15, y+30, 20, 10); // boca
}
```

## Exercício #2

- Modifique o Exemplo #3 de forma a que o método aceite mais um parâmetro: a largura da face

# Métodos

- Implementação

```
void nomeMetodo([<tipo> nomeParametro1, <tipo> nomeParametro2, ...]) {  
    [código do método]  
}
```

- Os parâmetros são variáveis que podem ser utilizadas no código do método

- Invocação

```
nomeMetodo(valorParametro1, valorParametro2, ...);
```

- O valor de cada parâmetro é atribuído automaticamente à variável correspondente quando o método é invocado

# Métodos

```
void desenhaCara(int x, int y, int largura)
```

- `desenhaCara(0, 0, 50);`

```
x = 0;  
y = 0;  
largura = 50;
```

```
fill(255, 255, 255);           //branco  
rect(x, y, largura, largura); //face
```

```
fill(0, 255, 0);               //verde  
rect(x+10, y+10, 10, 10);     //olho esquerdo  
rect(x+30, y+10, 10, 10);    //olho direito
```

```
fill(255, 0, 0);               //vermelho  
rect(x+15, y+30, 20, 10);    // boca
```

# Valor de Retorno

- Problema
  - Calcular a média dos valores de um vector

# Exemplo #4

```
float idadesT1[] = {10, 20, 30, 19, 15, 18};

void setup() {
    //calcular a media das cinco primeiras idades
    float soma = 0;
    float media = 0;
    for (int i = 0; i < 5; i++) {
        soma = soma + idadesT1[i];
    }
    media = soma/5;
    println(media);
}
```

# Valor de Retorno

- E se quiséssemos calcular a média de valores noutra vector?

# Exemplo #5

```
float idadesT1[] = {10, 20, 30, 19, 15, 18};
float idadesT2[] = {12, 23, 13, 17, 15, 28};
void setup() {
    //calcular a media das cinco primeiras idades do vector T1
    float soma = 0;
    float media = 0;

    for (int i = 0; i < 5; i++) {
        soma = soma + idadesT1[i];
    }
    media = soma/5;
    println(media);

    //calcular a media das cinco primeiras idades do vector T2
    soma = 0;
    media = 0;

    for (int i = 0; i < 5; i++) {
        soma = soma + idadesT2[i];
    }
    media = soma/5;
    println(media);
}
```

# Exemplo #6

```
float idadesT1[] = {10, 20, 30, 19, 15, 18};
float idadesT2[] = {12, 23, 13, 17, 15, 28};

void setup() {
    size(400, 400);

    //calcular a media das cinco primeiras idades do vector T1
    println(calculaMedia(idadesT1, 5));

    //calcular a media das cinco primeiras idades do vector T2
    println(calculaMedia(idadesT2, 5));
}

float calculaMedia(float idades[], float num) {
    float soma = 0;
    float media = 0;

    for (int i = 0; i < num; i++) {
        soma = soma + idades[i];
    }
    media = soma/num;
    return media;
}
```

# Métodos

- Implementação

```
<tipo> nomeMetodo([<tipo> nomeParametro1, <tipo> nomeParametro2, ...]) {  
    [código do método]  
}
```

- Métodos com <tipo> diferente de “void” retornam um valor

- estes métodos têm de ter a instrução

- `return <valor>;`

- “Void” é um tipo de retorno especial

- Significa “nada”

# Exercício #3

- Crie um método “circulo” com três parâmetros
  - X, Y e Raio
- O método deve desenhar um círculo e retornar
  - True – se o círculo foi desenhado completamente dentro do ecrã
  - False – se o círculo foi desenhado parcialmente fora do ecrã
- Use o método criado num programa que desenhe continuamente círculos no ecrã em posições e com raios diferentes
  - A cor do circulo depende se o anterior estava totalmente dentro do ecrã
    - Branco se o círculo anterior foi desenhado completamente dentro do ecrã
    - Cinzento se não.

# Visibilidade das Variáveis

- As variáveis podem ser declaradas em vários sítios do programa
  - no topo: programa principal
  - dentro dos métodos
  - dentro de blocos de código: { }
- O sítio onde são declaradas determina onde podem ser lidas/escritas

# Visibilidade das Variáveis

- Variáveis declaradas no corpo principal podem ser utilizadas em qualquer sítio
  - Como temos vindo a fazer até agora
- Variáveis declaradas dentro de métodos apenas podem ser lidas/escritas no próprio método
- Variáveis declaradas dentro de um bloco de código apenas podem ser lidas/escritas dentro desse bloco

# Projecto Semanal

- Crie um método que desenhe uma “flor”. Os parâmetros são as coordenadas  $x$  e  $y$  e a largura da flor. O caule deve ter 50 pixels de altura.
- Use o método anterior para “semear” 10 flores num “campo”.