

Desenvolvimento para Dispositivos Móveis

Introdução à DART

Profº: Joseph Donald

Contatos:

☎ (83) 98228-8607

📷 @josephdonald

✉ 030106382@uninassau.edu.br

“Se você tem uma maçã e eu tenho outra; e nós trocamos as maçãs, então cada um terá sua maçã. Mas se você tem uma ideia e eu tenho outra, e nós as trocamos; então cada um terá duas ideias.”

George Bernard Shaw

- Dart é uma linguagem de programação desenvolvida pelo Google que é utilizada para desenvolver aplicativos web e móveis.
- Linguagem principal do Flutter, um framework de desenvolvimento de aplicativos móveis.



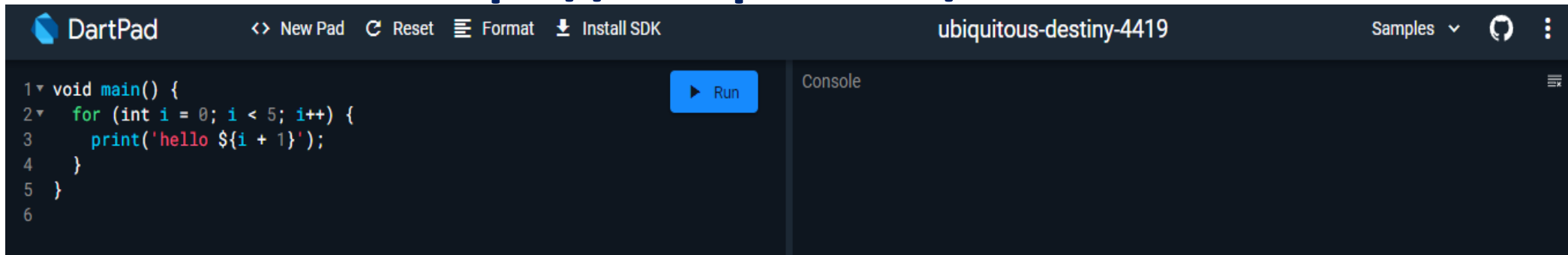
Dart



- A linguagem Dart é conhecida por sua sintaxe simples e fácil de aprender, que é semelhante à sintaxe de outras linguagens populares, como Java e JavaScript.
- Dart oferece recursos avançados, como suporte a tipos de dados fortemente tipados, programação orientada a objetos e gerenciamento de estados.



- Uma opção muito simples para aprender a linguagem é utilizar o site DartPad: <https://dartpad.dev/>

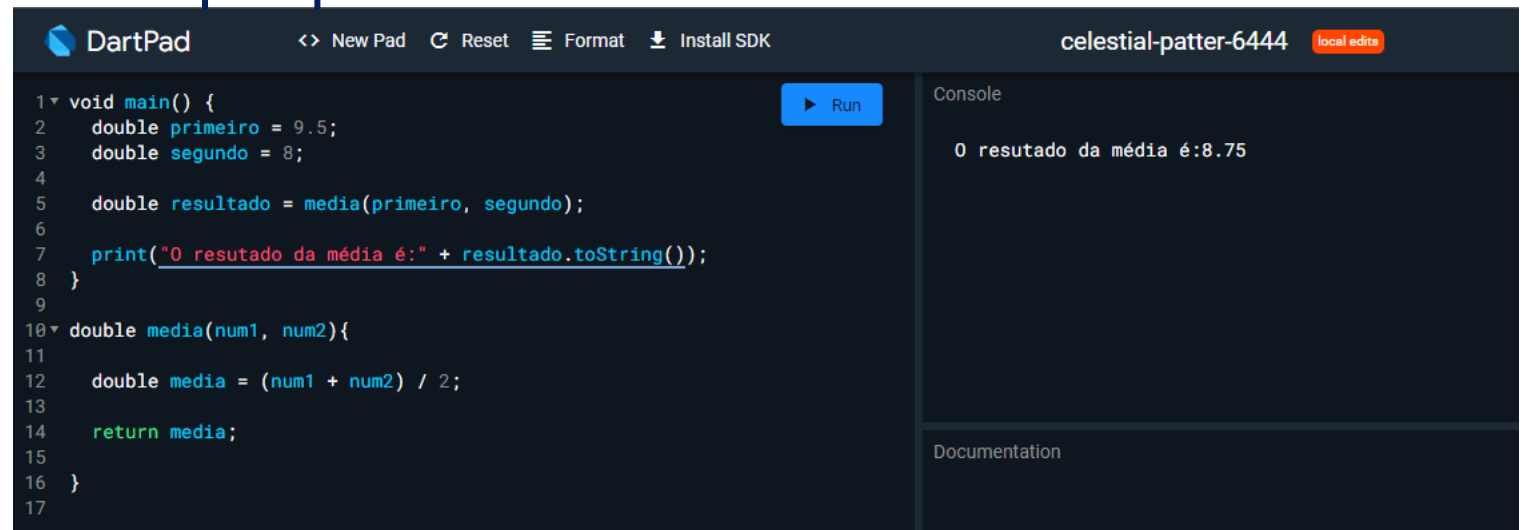


The screenshot shows the DartPad web interface. The top bar includes the DartPad logo, navigation links (New Pad, Reset, Format, Install SDK), the username 'ubiquitous-destiny-4419', and a 'Samples' dropdown. The main editor area contains the following Dart code:

```
1 void main() {  
2   for (int i = 0; i < 5; i++) {  
3     print('hello ${i + 1}');  
4   }  
5 }  
6
```

A 'Run' button is visible next to the code. The right sidebar shows a 'Console' section, which is currently empty.

Exemplo para cálculo de média:



The screenshot shows the DartPad web interface with a more complex Dart program. The top bar includes the DartPad logo, navigation links, the username 'celestial-patter-6444', and a 'local edit' button. The main editor area contains the following Dart code:

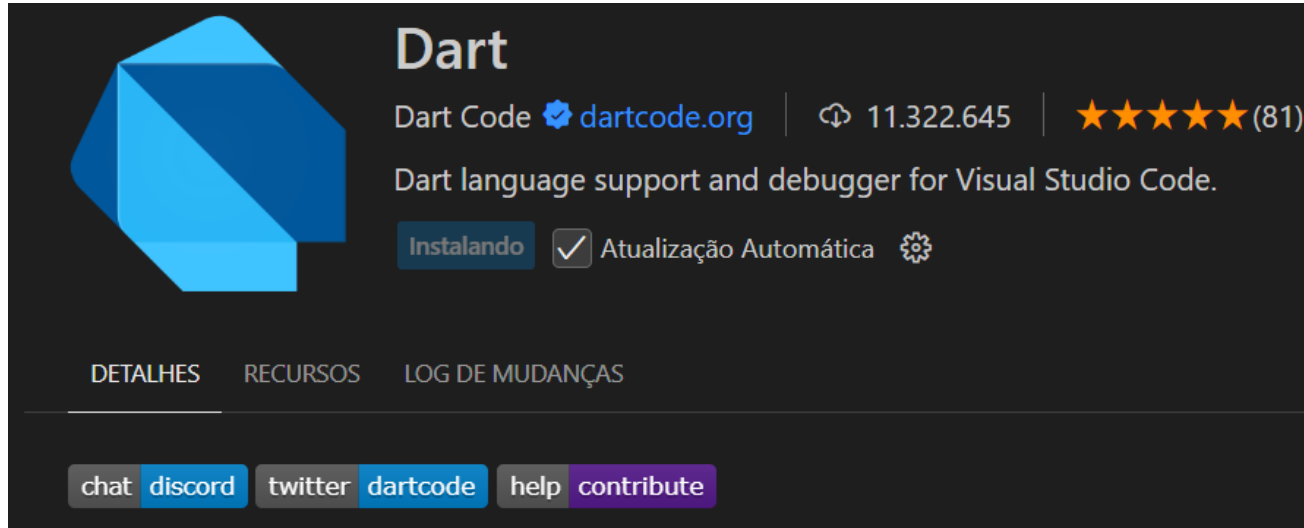
```
1 void main() {  
2   double primeiro = 9.5;  
3   double segundo = 8;  
4  
5   double resultado = media(primeiro, segundo);  
6  
7   print("O resultado da média é:" + resultado.toString());  
8 }  
9  
10 double media(num1, num2){  
11   double media = (num1 + num2) / 2;  
12  
13   return media;  
14 }  
15  
16  
17
```

A 'Run' button is visible next to the code. The right sidebar shows a 'Console' section with the output: 'O resultado da média é:8.75'. Below the console is a 'Documentation' section.





- Outra opção é realizando a instalação do SDK Dart/Flutter no S.O. e utilizando uma IDE de desenvolvimento, como o VSCode ou o Android Studio, por exemplo.



- Neste caso, além de instalar o SDK, você precisa ativar a extensão do DART no VSCode para compilar os códigos.



- A função **main()** é o ponto de entrada de um programa Dart. Ela é chamada automaticamente quando o programa é iniciado e é o primeiro código a ser executado. É a partir da função **main()** que podemos chamar outras funções e executar o código desejado.
- É uma parte fundamental de qualquer programa Dart e deve estar presente em todos os arquivos executáveis.

```
void main() {  
    print("Olá, mundo!");  
}
```

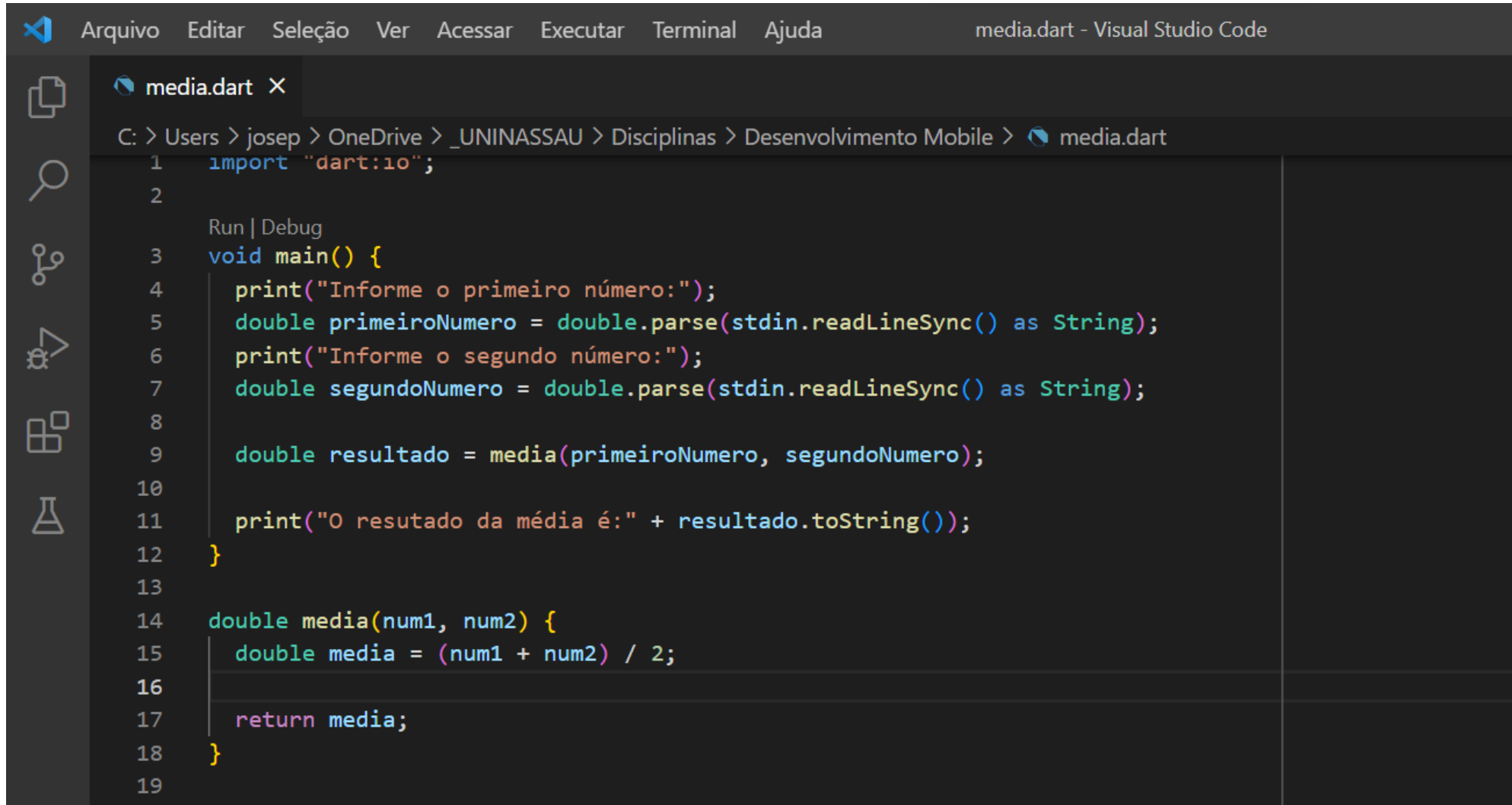


```
1  main() {  
2    print("### Meu primeiro código em DART ###");  
3    int ano = 2023;  
4    var texto = "Olá mundo!";  
5    print(texto + " Sou seu primeiro código DART em " + ano.toString() + "!!!");  
6  }  
7
```

```
C:\Users\josep\Desktop>dart ola_mundo.dart  
### Meu primeiro código em DART ###  
Olá mundo! Sou seu primeiro código DART em 2023!!!
```



Dart



```
1  import "dart:io";
2
3  void main() {
4      print("Informe o primeiro número:");
5      double primeiroNumero = double.parse(stdin.readLineSync() as String);
6      print("Informe o segundo número:");
7      double segundoNumero = double.parse(stdin.readLineSync() as String);
8
9      double resultado = media(primeiroNumero, segundoNumero);
10
11     print("O resutado da média é:" + resultado.toString());
12 }
13
14 double media(num1, num2) {
15     double media = (num1 + num2) / 2;
16
17     return media;
18 }
19
```


Abaixo, temos alguns tipos de variáveis utilizadas em Dart:

int: Variáveis do tipo inteiro.

Exemplo: `int idade = 30;`

double: Variáveis do tipo ponto flutuante.

Exemplo: `double preco = 12.99;`

String: Variáveis do tipo texto.

Exemplo: `String nome = 'Maria';`

bool: Variáveis do tipo booleano, que podem ter valores `true` ou `false`.

Exemplo: `bool temCarro = false;`

dynamic: Variáveis que podem ter qualquer tipo de valor.

Exemplo: `dynamic variavel = 10;` ou `dynamic variavel = 'texto';`

var: Variáveis que o tipo é inferido pelo compilador em tempo de execução.

Exemplo: `var valor = 10;` ou `var texto = 'algum texto';`

final: Variáveis cujo valor é definido apenas uma vez e não pode ser alterado.

Exemplo: `final PI = 3.14;` ou `final int FATOR_KM_MS = 3.6;`

const: Variáveis cujo valor é definido em tempo de compilação e não pode ser alterado. Exemplo: `const PI = 3.14;` ou `const int FATOR_KM_MS = 10;`

É importante escolher o tipo de variável correto para cada situação, para garantir que o código seja executado de forma eficiente e segura. Além disso, é uma boa prática sempre declarar as variáveis com um tipo explícito, para deixar o código mais claro e fácil de entender.

- Em Dart, você pode atribuir um valor a uma variável usando o operador de atribuição `=`. O tipo da variável pode ser declarado explicitamente ou pode ser inferido pelo compilador.
- Observe que os valores atribuídos devem ser compatíveis com o tipo da variável. Por exemplo, você não pode atribuir um valor inteiro a uma variável do tipo texto.

Aqui estão alguns exemplos de como atribuir valores a variáveis em Dart:

int idade = 30; // Declarando uma variável do tipo inteiro e atribuindo um valor

double preco = 12.99; // Declarando uma variável do tipo double e atribuindo um valor

String nome = 'Maria'; // Declarando uma variável do tipo texto e atribuindo um valor

bool temCarro = false; // Declarando uma variável do tipo booleano e atribuindo um valor

dynamic variavel = 10; // Declarando uma variável dinâmica e atribuindo um valor

var valor = 10; // Inferindo o tipo da variável a partir do valor atribuído

var texto = 'algum texto';



- Elabore um algoritmo em DART que solicite 2 números ao usuário, o software deverá efetuar as 4 operações básicas: soma, subtração, multiplicação e divisão, e ao final deve exibir o resultado de cada uma das operações.



DART oferece duas estruturas condicionais para controlar o fluxo de execução do código:

- “if/else”

```
int num = 10;  
if (num % 2 == 0) {  
    print("O número é par.");  
} else {  
    print("O número é ímpar.");  
}
```

- “switch/case”

```
String cor = "verde";  
switch (cor) {  
    case "vermelho":  
        print("A cor é vermelha.");  
        break;  
    case "azul":  
        print("A cor é azul.");  
        break;  
    case "verde":  
        print("A cor é verde.");  
        break;  
    default:  
        print("Não foi possível identificar a cor.");  
}
```



- Crie um programa em Dart que solicite a idade de uma pessoa e exiba sua categoria de acordo com a tabela abaixo. O desafio é utilizar **if** e **else** para a verificação.
- Criança de 0 a 13 anos
- Adolescente de 14 a 17 anos
- Adulto de 18 a 59 anos
- Idoso acima de 59 anos

DART oferece três tipos de estruturas de repetição para iterar sobre uma determinada lista de elementos:

- **for**
- **while**
- **do-while**

Estrutura de repetição “for”:

É usada quando se sabe a quantidade de iterações que serão necessárias. A sua sintaxe é composta de três partes: a inicialização de uma variável de controle; a condição que a variável de controle deve atender para a execução continuar; e o incremento ou decremento da variável de controle.



Exemplo de código:

```
for (int i = 0; i < 5; i++) {  
    print(i);  
}
```

Saída esperada:

0

1

2

3

4

Estrutura de repetição “while”:

É usada quando não se sabe a quantidade de iterações necessárias, mas se sabe uma condição que deve ser atendida para a execução continuar. A sua sintaxe é composta de uma condição que deve ser verdadeira para que o bloco de código dentro dela seja executado.



Exemplo de código:

```
int i = 0;  
while (i < 5) {  
    print(i);  
    i++;  
}
```

Saída esperada:

```
0  
1  
2  
3  
4
```

Estrutura de repetição “do-while”:

É semelhante a estrutura “while”, com a diferença de que o bloco de código dentro dela é executado pelo menos uma vez, mesmo que a condição não seja verdadeira.



Exemplo de código:

```
int i = 0;  
do {  
    print(i);  
    i++;  
} while (i < 5);
```

Saída esperada:

```
0  
1  
2  
3  
4
```

Em DART, temos três tipos de vetores:

- **Listas**
- **Conjuntos**
- **Mapas**



Listas:

- As listas são uma coleção ordenada de elementos do mesmo tipo, que podem ser acessados por um índice numérico.
- Os elementos podem ser adicionados, removidos e modificados a qualquer momento. A sua sintaxe é composta por colchetes [], e os elementos são separados por vírgulas.

```
List<String> frutas = ['maçã', 'banana', 'laranja'];  
print(frutas); // saída esperada: [maçã, banana, laranja]
```




Conjuntos:

- Os conjuntos são uma coleção não ordenada de elementos únicos do mesmo tipo, que não possuem índices numéricos e não permitem elementos duplicados.
- Os elementos podem ser adicionados e removidos a qualquer momento. A sua sintaxe é composta por chaves { }, e os elementos são separados por vírgulas.

```
Set<int> numeros = {1, 2, 3, 4, 5};  
print(numeros); // saída esperada: {1, 2, 3, 4, 5}
```



Mapas:

- Os mapas são uma coleção de pares chave-valor, onde cada chave é única e é usada para acessar um valor associado.
- Os elementos podem ser adicionados, removidos e modificados a qualquer momento. A sua sintaxe é composta por chaves {}, e os pares chave-valor são separados por vírgulas e dois pontos :

```
Map<String, String> telefones = {'João': '123456', 'Maria': '654321', 'Pedro': '789456'};  
print(telefones); // saída esperada: {João: 123456, Maria: 654321, Pedro: 789456}
```



- Para criar funções em Dart, usamos a palavra-chave `void` para indicar que a função não retorna nenhum valor, ou um tipo de retorno caso a função retorne algum valor.
- Em seguida, escrevemos o nome da função, seguido de parênteses que podem conter parâmetros de entrada separados por vírgulas. Por fim, escrevemos o corpo da função, delimitado por chaves `{ }`. O corpo da função contém o código que será executado quando a função for chamada.



- Função com retorno

```
int soma(int a, int b) {  
    int resultado = a + b;  
    return resultado;  
}
```

- Função sem retorno

```
void saudacao(String nome) {  
    print("Olá, $nome! Como você está?");  
}
```



- Elabore um algoritmo em DART que calcule o IMC de dona Clarabela. Estes são os dados dela:
 - Altura: 1,63
 - Peso: 45 kg
- Depois de encontrar o índice de massa corporal, o algoritmo deve classificar a condição física de dona Clarabela de acordo com a seguinte tabela:

Condição	Situação
IMC abaixo de 20	Abaixo do peso
IMC de 20 até 25	Peso Normal
IMC de 25 até 30	Sobre Peso
IMC de 30 até 40	Obeso
IMC de 40 e acima	Obeso Mórbido



UNINASSAU

Desenvolvimento Mobile
Profº: Joseph Donald

SOLUÇÃO: