Algoritmos e Estruturas de Dados

Funções de entrada e saída

Java

Funções de entrada e saída

As funções de entrada e saída realizam a conexão do algoritmo com o ambiente externo, recebendo e enviando dados.

Saída de dados

A saída de dados que utilizaremos aqui é a tela. Para mostrar uma mensagem na tela, utilizaremos a função abaixo, acompanhada da mensagem que desejamos apresentar na tela.

```
System.out.println();
```

Exemplo

- Para apresentar na tela a mensagem `Olá Mundo`:

```
System.out.println("Olá Mundo");
```

Saída na tela:

```
Olá Mundo
```

Atenção

Perceba que aqui estamos observando apenas a função de saída. Para que ela possa funcionar de maneira correta no algoritmo, é necessário que ela esteja dentro de uma **estrutura básica**.

```
public class Main{
  public static void main(String args){
    System.out.println("Olá Mundo");
  }
}
```

Saída na tela:

```
Olá Mundo
```

Exemplo

- Para apresentar a mensagem `Bom dia, José`:

```java
System.out.println("Bom dia, José");
```

Saída na tela:

```
Bom dia, José
```

Exercício

- Desenvolva um algoritmo que apresenta seu nome na saída padrão.
- Para apresentar um valor específico:

```
System.out.println(18);
```

Saída na tela:

```
18
```

- Pode-se também apresentar diversos resultados de uma única vez:

```
System.out.println("A idade mínima deve ser de " + 18 + " anos");
```

Saída na tela:

```
A idade mínima deve ser de 18 anos
```

Uso de armazenamento intermediário

No desenvolvimento de algoritmos, podemos utilizar armazenamento intermediário de valores. Para tal, devemos indicar qual é o **tipo de dado** que deverá ser guardado e um **identificador** desse armazenamento.

Informação

O uso desse armazenamento é um tópico de grande importância no desenvolvimento de algoritmos. O veremos de forma mais detalhada ao abordarmos tipos primitivos de dados, constantes e variáveis.

Exemplo

- Armazenando um valor e apresentando na tela

```
int altura;
altura = 50; //identificador que permite armazenar um número inteiro
System.out.println(altura); //saída: 50
```

Saída na tela:

```
50
```

É possível também apresentar diversos valores de uma única vez em uma única saída.

Exemplo

```
String nome = "Dunga";
int idade = 35;
System.out.println(nome + " tem " + anos " de idade.");
```

Saída na tela:

```
Dunga tem 35 anos de idade
```

Exercícios

1. Desenvolva um algoritmo que mostra na tela a mensagem *Hoje é dia de aula de Algoritmos*.

2. Desenvolva um algoritmo que dentro dele possui um armazenamento intermediário de caracteres chamado *mes_ferias_1* e *mes_ferias_2*, onde cada um deve possuir os valores *janeiro* e *julho*. Seu algoritmo deve informar na tela uma mensagem informando que estes são os meses de férias previsto no calendário acadêmico.

3. Desenvolva um algoritmo que dentro dele possui os armazenamentos intermediários de caracteres chamado *mes_ferias_1* e *mes_ferias_2*, onde cada um deve possuir os valores *janeiro* e *julho*. Também possui os armazenamentos intermediários *dias_ferias_1* e *dias_ferias_2*, com os conteúdos *30* e *15*, respectivamente. Seu algoritmo deve informar na tela uma mensagem informando que são previstos 30 dias de férias em janeiro e 15 dias de férias em julho.

Entrada de dados

Para que a entrada de dados possa ser realizado é necessário passar ao algoritmo uma informação adicional, com o identificador em que o valor deverá ser armazenado.

Para a entrada de dados pode-se utilizar a seguinte função:

Antes de iniciar a leitura de dados, é necessário associar um identificador à entrada.

```
Scanner entrada = new Scanner(System.in); //entrada é um identificador
```

Em seguida é possível ler o dado desejado utilizando `.next()` ou `.nextInt()`

```
entrada.next(); //para dados do tipo caractere
```

ou

```
entrada.nextInt(); //para dados do tipo inteiro
```

Atenção

Para que a função de entrada possa ser utilizada é necessário importar a biblioteca também é necessário importar a biblioteca `Scanner` .

Para tal, é necessário incluir no início do código a linha

```
import java.util.Scanner;
```

Atenção

O código `Scanner entrada = new Scanner(System.in);` realiza uma associação do identificador `entrada` com a entrada padrão (comumente, o teclado).

Após a finalização das entradas, é necessário desassociar o identificador, com a função `entrada.close()`.

Conhecendo a função que realiza a leitura de dados da entrada padrão, devemos informar qual identificador será responsável por armazenar o dado recebido na entrada.

Exemplo

- recebendo valores na entrada e armazenando

recebendo da entrada um dado do tipo inteiro:

```
int numero;
Scanner entrada = new Scanner(System.in);
numero = entrada.nextInt();
entrada.close();
```

recebendo da entrada um dado do tipo caractere:

```
String palavra;
Scanner entrada = new Scanner(System.in);
palavra = entrada.next();
entrada.close();
```

Exemplo

- Solicite ao usuário que digite seu nome

```
//identificadores
String nome;

//entrada
// highlight-next-line
nome = entrada.next()   //recebe o dado da entrada
```

<details> <summary>Código completo</summary>

```
public class Main{
  public static void main(String args){
    //identificadores
    String nome;
    Scanner entrada;
```

Exemplo

- Pergunte ao usuário seu nome e idade. Em seguida, mostre na tela as informações digitadas.

```
//Identificadores
String nome;
int idade;

//Entrada
System.out.println("Nome: ");
nome = entrada.next();
System.out.println("Idade: ");
idade = entrada.nextInt();

//Saída
System.out.prinln(nome + " tem " + idade + " anos de idade.");
```

Exercícios

1. Solicite ao usuário que digite um número. Em seguida, mostre na tela o número digitado.
2. Desenvolva um algoritmo que pergunta ao usuário o nome do usuário, e em seguida responde "Boa noite, `user`", substituindo `user` pelo nome digitado.
3. Faça um algoritmo que pergunta ao usuário a sua idade, e em seguida informa a mensagem "Você tem `X` anos", substituindo `X` pela idade digitada.