Funções e Procedimentos

Prof. Tiago Gonçalves Botelho

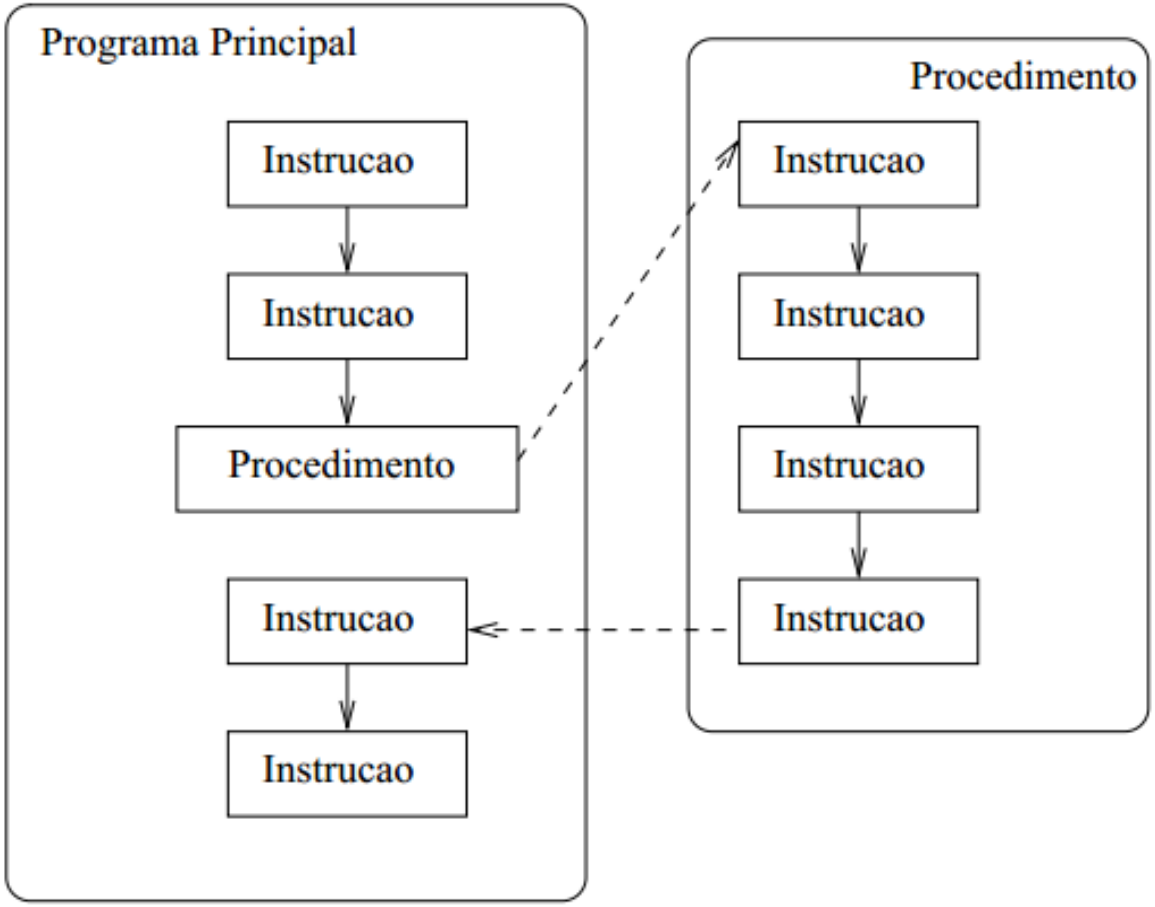
Funções e Procedimentos (Modularização)

- A modularização de um algoritmo/programa é uma forma de dividir as tarefas em subalgoritmos/subprogramas, e cada um desses módulos cuida de uma parte separada do problema.

Motivação para o uso de Funções e Procedimentos

- Permitir o reaproveitamento de código já construído (por você ou por outros programadores);
- Evitar que um trecho de código que seja repetido várias vezes dentro de um mesmo programa;
- Permitir a alteração de um trecho de código de uma forma mais rápida. Com o uso de uma função é preciso alterar apenas dentro da função que se deseja;
- Evitar que os blocos do programa fiquem grandes demais e, por consequência, mais difíceis de entender;

Fluxo de execução de um programa usando Procedimento



Definição de procedimentos e funções

- **Procedimentos** são estruturas que agrupam um conjunto de comandos, que são executados quando o procedimento é chamado.

- **Funções** são semelhantes aos procedimentos, exceto que uma função sempre retorna um valor.

Funções e Procedimentos

- Funções que não retornam valor **(Procedimentos)**
  - Sem parâmetro
  - Com parâmetro
- Funções que retornam valor **(Funções)**
  - Sem parâmetro
  - Com parâmetro
- Variável global (pode ser utilizada em todo o programa) e Variável local (só é válida dentro do procedimento ou função)

Funções que não retornam valor e sem parâmetro

- Exemplo 01: Faça um programa que receba dois valores no programa principal, some os dois números em um procedimento e exiba o resultado na função principal.

Funções que não retornam valor e sem parâmetro

Passo 1: Ler os dados de entrada na função principal, de acordo com o que o enunciado pede.

```
 1  #include<bits/stdc++.h>
 2
 3
 4
 5
 6  int main(){
 7      printf("Entre com um valor: ");
 8      scanf("%d",&a);
 9      printf("Entre com outro valor: ");
10      scanf("%d",&b);
11
12
13      return 0;
14  }
15
```

Funções que não retornam valor e sem parâmetro

Passo 2: Fazer a chamada da função, o início de sua implementação e o protótipo da função (antes do main).

```
1   #include<bits/stdc++.h>
2
3
4   void soma(void);
5
6   int main(){
7       printf("Entre com um valor: ");
8       scanf("%d",&a);
9       printf("Entre com outro valor: ");
10      scanf("%d",&b);
11      soma();
12
13      return 0;
14  }
15
16  void soma(void){
17
18  }
19
```

Não retorna valor

Sem parâmetro

Funções que não retornam valor e sem parâmetro

Passo 3: Implementar a operação na função e mostrar o resultado na função principal conforme o enunciado solicita.

```
 1  #include<bits/stdc++.h>
 2
 3
 4  void soma(void);
 5
 6  int main(){
 7      printf("Entre com um valor: ");
 8      scanf("%d",&a);
 9      printf("Entre com outro valor: ");
10      scanf("%d",&b);
11      soma();
12      printf("A soma e: %d",s);
13      return 0;
14  }
15
16  void soma(void){
17      s=a+b;
18  }
19
```

Funções que não retornam valor e sem parâmetro

Passo 4: Observar e declarar as variáveis como local e/ou global.

```
1   #include<bits/stdc++.h>
2
3   int a,b,s;
4   void soma(void);
5
6   int main(){
7       printf("Entre com um valor: ");
8       scanf("%d",&a);
9       printf("Entre com outro valor: ");
10      scanf("%d",&b);
11      soma();
12      printf("A soma e: %d",s);
13      return 0;
14  }
15
16  void soma(void){
17      s=a+b;
18  }
```

Funções que não retornam valor e com parâmetro

- Exemplo 02: Faça um programa que leia um valor representando os segundos no programa principal. Depois passe-o como parâmetro para um procedimento, que deverá convertê-lo para horas, minutos e segundos.

Funções que não retornam valor e com parâmetro

Passo 1: Ler o dado de entrada na função principal, de acordo com o que o enunciado pede.

```
1   #include<bits/stdc++.h>
2
3
4
5   int main(){
6
7       printf("Entre com os segundos: ");
8       scanf("%d",&seg);
9
10      return 0;
11  }
12
13
```

Funções que não retornam valor e com parâmetro

Passo 2: Fazer a chamada da função, o início de sua implementação e o protótipo da função (antes do main).

```
1   #include<bits/stdc++.h>
2
3   void transformacao(int);
4
5   int main(){
6
7       printf("Entre com os segundos: ");
8       scanf("%d",&seg);
9       transformacao(seg);
10      return 0;
11  }
12
13  void transformacao(int segundos){
14
15  }
16
```

Não retorna valor

Com parâmetro

Funções que não retornam valor e com parâmetro

Passo 3: Implementar a operação e mostrar o resultado na função conforme o enunciado solicita.

```
1  #include<bits/stdc++.h>
2
3  void transformacao(int);
4
5  int main(){
6
7      printf("Entre com os segundos: ");
8      scanf("%d",&seg);
9      transformacao(seg);
10     return 0;
11 }
12
13 void transformacao(int segundos){
14
15     h=segundos/3600;
16     r=segundos%3600;
17     m=r/60;
18     s=r%60;
19     printf("Hora=%d\nMinutos=%d\nSegundos=%d\n",h,m,s);
20 }
```

Funções que não retornam valor e com parâmetro

Passo 4: Observar e declarar as variáveis como local e/ou global.

```
1   #include<bits/stdc++.h>
2
3   void transformacao(int);
4
5   int main(){
6       int seg;
7       printf("Entre com os segundos: ");
8       scanf("%d",&seg);
9       transformacao(seg);
10      return 0;
11  }
12
13  void transformacao(int segundos){
14      int h,m,s,r;
15      h=segundos/3600;
16      r=segundos%3600;
17      m=r/60;
18      s=r%60;
19      printf("Hora=%d\nMinutos=%d\nSegundos=%d\n",h,m,s);
20  }
```

Funções que retornam valor e sem parâmetro

- Exemplo 03: Faça um programa que receba dois valores na função principal. A seguir utilize uma função para calcular a multiplicação de dois números e retorne o resultado para mostrar no programa principal.

Funções que retornam valor e sem parâmetro

Passo 1: Ler os dados de entrada na função principal, de acordo com o que o enunciado pede.

```
1   #include<bits/stdc++.h>
2
3
4   int main(){
5
6       printf("Entre com um valor: ");
7       scanf("%d",&x);
8       printf("Entre com outro valor: ");
9       scanf("%d",&y);
10
11
12      return 0;
13  }
```

Funções que retornam valor e sem parâmetro

Passo 2: Fazer a chamada da função (preparar uma variável para receber o retorno da função), o início de sua implementação e o protótipo da função (antes do main).

```
1   #include<bits/stdc++.h>
2
3   int multiplica(void);
4
5
6   int main(){
7
8       printf("Entre com um valor: ");
9       scanf("%d",&x);
10      printf("Entre com outro valor: ");
11      scanf("%d",&y);
12      res=multiplica();
13
14      return 0;
15  }
16
17  int multiplica(void){
18
19  }
```

Retorna valor

Sem parâmetro

Funções que retornam valor e sem parâmetro

Passo 3: Implementar a operação e mostrar o resultado na função conforme o enunciado solicita.

```
1   #include<bits/stdc++.h>
2
3   int multiplica(void);
4
5
6   int main(){
7
8       printf("Entre com um valor: ");
9       scanf("%d",&x);
10      printf("Entre com outro valor: ");
11      scanf("%d",&y);
12      res=multiplica();
13      printf("A multiplicacao vale: %d\n",res);
14      return 0;
15  }
16
17  int multiplica(void){
18
19      r=x*y;
20      return r;
21  }
```

Funções que retornam valor e sem parâmetro

Passo 4: Observar e declarar as variáveis como local e/ou global.

```
1  #include<bits/stdc++.h>
2
3  int multiplica(void);
4  int x,y;
5
6  int main(){
7      int res;
8      printf("Entre com um valor: ");
9      scanf("%d",&x);
10     printf("Entre com outro valor: ");
11     scanf("%d",&y);
12     res=multiplica();
13     printf("A multiplicacao vale: %d\n",res);
14     return 0;
15 }
16
17 int multiplica(void){
18     int r;
19     r=x*y;
20     return r;
21 }
22
```

Funções que retornam valor e com parâmetro

- Exemplo 04: Alterar o exercício anterior para passar como parâmetro os dois valores a serem multiplicados e retornar valor.

Funções que retornam valor e com parâmetro

Passo 1: Ler os dados de entrada na função principal, de acordo com o que o enunciado pede.

```
 1  #include<bits/stdc++.h>
 2
 3
 4
 5  int main(){
 6
 7      printf("Entre com um valor: ");
 8      scanf("%d",&x);
 9      printf("Entre com outro valor: ");
10      scanf("%d",&y);
11
12
13      return 0;
14  }
```

Funções que retornam valor e com parâmetro

Passo 2: Fazer a chamada da função (preparar uma variável para receber o retorno da função), o início de sua implementação e o protótipo da função (antes do main).

```
1   #include<bits/stdc++.h>
2
3   int multiplica(int,int);
4
5   int main(){
6
7       printf("Entre com um valor: ");
8       scanf("%d",&x);
9       printf("Entre com outro valor: ");
10      scanf("%d",&y);
11      res=multiplica(x,y);
12
13      return 0;
14  }
15
16  int multiplica(int a, int b){
17
18  }
```

Retorna valor

Com parâmetros

Funções que retornam valor e com parâmetro

Passo 3: Implementar a operação e mostrar o resultado na função conforme o enunciado solicita.

```
1   #include<bits/stdc++.h>
2
3   int multiplica(int,int);
4
5   int main(){
6
7       printf("Entre com um valor: ");
8       scanf("%d",&x);
9       printf("Entre com outro valor: ");
10      scanf("%d",&y);
11      res=multiplica(x,y);
12      printf("A multiplicacao vale: %d\n",res);
13      return 0;
14  }
15
16  int multiplica(int a, int b){
17
18      r=a*b;
19      return r;
20  }
```

Funções que retornam valor e com parâmetro

Passo 4: Observar e declarar as variáveis como local e/ou global.

```
1  #include<bits/stdc++.h>
2
3  int multiplica(int,int);
4
5  int main(){
6      int x,y,res;
7      printf("Entre com um valor: ");
8      scanf("%d",&x);
9      printf("Entre com outro valor: ");
10     scanf("%d",&y);
11     res=multiplica(x,y);
12     printf("A multiplicacao vale: %d\n",res);
13     return 0;
14 }
15
16 int multiplica(int a, int b){
17     int r;
18     r=a*b;
19     return r;
20 }
21
```

Referências Bibliográficas

- ASCENCIO, A. F. G.; CAMPOS, E. A. V. Fundamentos da programação de computadores. Editora Prentice-Hall, 2008.
- MEDINA, M.; FERTIG, C. Algoritmos e Programação: teoria e prática. 2ª ed.São Paulo: Editora Novatec, 2006.