Templates

Prof. Thiago Felski Pereira, MSc.
Adaptado: Elisangela Maschio de Miranda

Templates

- Permite criar funções genéricas (podendo ser considerados Generalizadores de Tipos) que admitem qualquer tipo de dado como parâmetro e retorna um valor sem ter que sobrecarregar a função com todos os tipos de dados possíveis.
- Isso é muito útil para definir funções que podem receber **parâmetros de tipos diferentes, mas que possuam lógicas de programação idênticas**.
- Os parâmetros de tipo formais são tipos primitivos ou tipos definidos pelo programador, usados para especificar os tipos dos parâmetros da função.

Templates

```
1   #include <iostream>
2   using namespace std;
3   #include <locale.h>
4
5   template <typename Tipo>
6   Tipo soma (Tipo a, Tipo b) {
7       return (a + b);
8   }
9
10  int main() {
11      setlocale(LC_ALL, "Portuguese");
12
13      cout<<endl<<"Soma de dois int: "<<soma(2 , 4);
14      cout<<endl<<"Soma de dois float: "<<soma(2.3 , 7.6);
15      cout<<endl<<"Soma de dois caracteres: "<<soma('1' , 'x');
16
17      return 0;
18  }
```

```
C:\Users\Usuario\Desktop\Teste\bin\Debug\Teste.exe

Soma de dois int: 6
Soma de dois float: 9.9
Soma de dois caracteres: ©
Process returned 0 (0x0)   execution time : 0.019 s
Press any key to continue.
```

Templates

```
1   #include <iostream>
2   using namespace std;
3   #include <locale.h>
4
5   template <typename Tipo>
6   Tipo maior (Tipo a, Tipo b) {
7       return (a>b? a : b);
8   }
9
10  int main() {
11      setlocale(LC_ALL, "Portuguese");
12
13      cout<<endl<<"Maior entre dois int: "<<maior(2 , 4);
14      cout<<endl<<"Maior entre dois float: "<<maior(2.3 , 7.6);
15      cout<<endl<<"Maior entre dois caracteres: "<<maior('x' , 'c');
16
17      return 0;
18  }
```

```
C:\Users\Usuario\Desktop\Teste\bin\Debu

Maior entre dois int: 4
Maior entre dois float: 7.6
Maior entre dois caracteres: x
Process returned 0 (0x0)   execution time : 0.021 s
Press any key to continue.
```

Templates

```
1  #include <iostream>
2  using namespace std;
3  #include <locale.h>
4  #define TAM 5
5
6  template <typename X>
7  void leituraVetor (X vetor[TAM]) {
8      int i;
9      for (i=0; i<TAM; i++) {
10         cout<<"Vetor na posição "<<i<<": ";
11         cin>>vetor[i];
12     }
13 }
14
15 template <typename X>
16 void escritaVetor (X vetor[TAM]) {
17     int i;
18     cout<<endl<<endl;
19     for (i=0; i<TAM; i++) {
20         cout<<vetor[i]<<"\t";
21     }
22 }
```

```
23
24 int main() {
25     setlocale(LC_ALL, "Portuguese");
26
27     int vetInt[TAM];
28     leituraVetor(vetInt);
29     escritaVetor(vetInt);
30
31     cout<<endl<<endl;
32     string vetChar[TAM];
33     leituraVetor(vetChar);
34     escritaVetor(vetChar);
35     return 0;
36 }
```

```
C:\Users\Usuario\Desktop\Teste\bin\Debu
Vetor na posição 0: 1
Vetor na posição 1: 2
Vetor na posição 2: 3
Vetor na posição 3: 4
Vetor na posição 4: 5


1       2       3       4       5

Vetor na posição 0: Agua
Vetor na posição 1: Terra
Vetor na posição 2: Fogo
Vetor na posição 3: Ar
Vetor na posição 4: Coracao


Agua    Terra   Fogo    Ar      Coracao
Process returned 0 (0x0)   execution time : 11.073 s
Press any key to continue.
```


UNIVALI

Obrigado pela atenção

contato: Felski@univali.br