Типове II: Шаблони на функции. Указатели към функции

Калин Георгиев

15 януари 2016 г.

Шаблони (на функции)

Еднообразни функции за различни типове

```
int findIndexMax
    (int arr[], int arrSize)
{
  int indexMax = 0;
  for (int i = 1; i < arrSize; i++)
    if (arr[indexMax] < arr[i])
      indexMax = i;

  return indexMax;
}
int findIndexMax
    (char arr[], int arrSize)
{
  int indexMax = 0;
  for (int i = 1; i < arrSize; i++)
    if (arr[indexMax] < arr[i])
      indexMax = i;

  return indexMax;
}
```

```
int findIndexMax
    (double arr[], int arrSize)
{
  int indexMax = 0;
  for (int i = 1; i < arrSize; i++)
    if (arr[indexMax] < arr[i])
      indexMax = i;

  return indexMax;
}

...
```

Създаване на "Шаблон на функция"

```
template <typename T>
int findIndexMax  (T arr[], int arrSize)
{
  int indexMax = 0;
  for (int i = 1; i < arrSize; i++)
    if (arr[indexMax] < arr[i])
      indexMax = i;

  return indexMax;
}
```

Използване на шаблона на функция

```
int main ()
{
  int arri[] = {1,5,6,7};
  cout << findIndexMax<int> (arri,4);

  double arrd[] = {2.1,17.5,6.0};
  cout << findIndexMax<double> (arrd,3);

  char arrc[] = "Hello";
  cout << findIndexMax<char> (arrc,5);

  char* arrstr[] = {"Hello", "World", "!"};
  cout << findIndexMax<char*> (arrstr,3); //!!!
}
```

```
template <typename T>
int findIndexMax
   (T arr[], int arrSize)
{
  int indexMax = 0;
  for (int i = 1; i < arrSize; i++)
    if (arr[indexMax] < arr[i])
      indexMax = i;

  return indexMax;
}
```

- Конкретният тип трябва да е съвместим с всички операции с него, които се прилагат в шаблона (в горния пример - $<$)!

Още един пример: печатане на "всякакви" масиви

```
template <typename T>
void printArray  (T arr[], int arrSize)
{
  cout << "{"
  for (int i = 0; i < arrSize-1; i++)
    cout << arr[i] << ",";

  if (arrSize > 0) //no comma
    cout << arr[arrSize-1];

  cout << "}";
}
```

Още по-дълбока параметризация: функции като параметри

Пример за еднотипни фунцкии

```
int findIndexMax  (int arr[], int arrSize)
{
  int index = 0;
  for (int i = 1; i < arrSize; i++)
    if (arr[index] < arr[i])
      index = i;

  return index;
}

int findIndexMin  (int arr[], int arrSize)
{
  int index = 0;
  for (int i = 1; i < arrSize; i++)
    if (arr[index] > arr[i])
      index = i;

  return index;
}
```

Функции вместо операторите < и >

```
bool compareGt (int a, int b)
{
  return a > b;
}
bool compareLt (int a, int b)
{
  return a < b;
}
```

```
int findIndexMax
   (int arr[], int arrSize)
{
  int index = 0;
  for (int i = 1; i < arrSize; i++)
    if (compareLt (arr[index],arr[i]))
      index = i;

  return index;
}

int findIndexMin
   (int arr[], int arrSize)
{
  int index = 0;
  for (int i = 1; i < arrSize; i++)
    if (compareGt (arr[index],arr[i]))
      index = i;

  return index;
}
```

Функциите имат тип

$$comparator : int \times int \rightarrow bool$$

```
bool compareGt (int a, int b)
{return a > b;}
bool compareLt (int a, int b)
{return a < b;}
```

$$ptrFn : T_1 \times T_2 \times ... \times T_k \rightarrow T_{res}$$

```
Tres (*ptrFn) (T1,T2,...,Tk);
```

```
int main (){
  //variable definition:
  //pComparator
  bool (*pComparator) (int,int);

  //pointer assignment
  pComparator = compareLt;
  cout << pComparator (1,2);

  pComparator = compareGt;
  cout << pComparator (1,2);
}
```

Функциите имат тип

$$comparator : int \times int \rightarrow bool$$

```
bool compareGt (int a, int b)
{return a > b;}
bool compareLt (int a, int b)
{return a < b;}
```

$$ptrFn : T_1 \times T_2 \times ... \times T_k \rightarrow T_{res}$$

```
Tres (*ptrFn) (T1,T2,...,Tk);
```

```
int main (){
  //variable definition:
  //pComparator
  bool (*pComparator) (int,int);

  //pointer assignment
  pComparator = compareLt;
  cout << pComparator (1,2);

  pComparator = compareGt;
  cout << pComparator (1,2);
}
```

Функциите имат тип

$$comparator : int \times int \rightarrow bool$$

```
bool compareGt (int a, int b)
{return a > b;}
bool compareLt (int a, int b)
{return a < b;}
```

$$ptrFn : T_1 \times T_2 \times ... \times T_k \rightarrow T_{res}$$

```
Tres (*ptrFn) (T1,T2,...,Tk);
```

```
int main (){
  //variable definition:
  //pComparator
  bool (*pComparator) (int,int);

  //pointer assignment
  pComparator = compareLt;
  cout << pComparator (1,2);

  pComparator = compareGt;
  cout << pComparator (1,2);
}
```

Функциите имат тип

```
bool compareGt (int a, int b)
{return a > b;}
bool compareLt (int a, int b)
{return a < b;}
```

$$comparator : int \times int \rightarrow bool$$

$$ptrFn : T_1 \times T_2 \times ... \times T_k \rightarrow T_{res}$$

```
Tres (*ptrFn) (T1,T2,...,Tk);
```

```
int main (){
  //variable definition:
  //pComparator
  bool (*pComparator) (int,int);

  //pointer assignment
  pComparator = compareLt;
  cout << pComparator (1,2);

  pComparator = compareGt;
  cout << pComparator (1,2);
}
```

Предаване на функции като параметри

```
int findExtremum
    (int arr[],
     int arrSize,
     bool (*pComparator)(int,int))
{
  int index = 0;
  for (int i = 1; i < arrSize; i++)
    if (pComparator (arr[index],arr[i]))
      index = i;

  return index;
}
```

```
bool compareGt (int a, int b)
{return a > b;}
bool compareLt (int a, int b)
{return a < b;}
```

Предаване на функции като параметри

```
void sort (int arr[],
           int arrSize,
           bool (*pComparator)(int,int))
{
  for (int i = 0; i < arrSize-1; i++)
  {
    //find subarray extremum and
    //swap with a[i]
    swap (arr[i],
          arr[i+findExtremum(arr+i,
                             arrSize-i,
                             pComparator)]);
  }

}
```

Предаване на функции като параметри

```
int main ()
{
  int arr[] = {1,7,3,5,2,3,2,4};

  sort (arr,8,compareLt);
  printArray (arr,8);

  sort (arr,8,compareGt);
  printArray (arr,8);

  return 0;
}
```

```
bool compareGt (int a, int b)
{return a > b;}
bool compareLt (int a, int b)
{return a < b;}

int findExtremum
   (int arr[],
    int arrSize,
    bool (*pComparator)(int,int))
{
  int indexMax = 0;
  for (int i = 1; i < arrSize; i++)
    if (pComparator (arr[indexMax],arr[i]))
      indexMax = i;
  return indexMax;
}


void sort (int arr[],
           int arrSize,
           bool (*pComparator)(int,int))
{
  for (int i = 0; i < arrSize-1; i++)
  {
    swap (arr[i],
          arr[i+findExtremum(arr+i,
                             arrSize-i,
                             pComparator)]);
  }
}
```

Благодаря за вниманието!