Типове III: Предефиниране на типове. Шаблони на указатели към функции

Калин Георгиев

21 май 2020 г.

Внимание: **-std=c++11**

Предефиниране на типове

Предефиниране на типове

- Използване на “сложен” тип:

```
void doSomething (int myMatrix[10][20])

int main ()
{
  int m[10][20] = {...};
  doSomething (m);
}
```

Предефиниране на типове

- "Полагане" на ново име на тип:

```
using myarr = int[10][20];

void doSomething (myarr myMatrix)

int main ()
{
  myarr m = {...};
  doSomething (m);

  //???
  myarr x[10];
}
```

```
void doSomething
     (int myMatrix[10][20])

int main ()
{
  int m[10][20] = {...};
  doSomething (m);
}
```

Тип на указател към фунцкия

Указател към функция

```
using Comparator =
    bool (*)(int,int);
```

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

$comparator : int \times int \rightarrow bool$

Предаване на функции като параметри

```
using Comparator = bool (*)(int,int);

//int findExtremum (int arr[],
//                  int arrSize,
//                  bool (*pComparator)(int,int));

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

  return indexMax;
}
```

Шаблони на указатели към функции

Шаблон на указател

```
template <typename T>
using Comparator = bool (*)(T,T);

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

  return indexMax;
}
```

```
//int findExtremum
//    (int arr[],
//     int arrSize,
//     bool (*pComparator)(int,int));

using Comparator = bool (*)(int,int);

int findExtremum (int arr[],
                  int arrSize,
                  Comparator pComparator);
```

Пример

```
template <typename T>
bool compareGt (T a, T b)
{return a > b;}

bool compareGt (char a, char b)
{return a < b;}

template <typename T>
bool compareLt (T a, T b)
{return a < b;}

int main ()
{
  int ia[] = {1,3,5};
  double da[] = {1.7,6.5,3.4,5.8};
  char ca = "abz";

  cout << findExtremum<int> (ia,3,compareGt<int>);
  cout << findExtremum<double> (da,4,compareGt<double>);
  cout << findExtremum<char> (ca,3,compareGt<char>);
```

```
template <typename T>
using Comparator = bool (*)(T,T);

template <typename T>
int findExtremum (T arr[],
                  int arrSize,
                  Comparator<T> pComparato
{
  int indexMax = 0;
  for (int i = 1; i < arrSize; i++)
    if (pComparator (arr[indexMax],arr[i])
      indexMax = i;

  return indexMax;
}
```

Благодаря за вниманието!