ШАБЛОНИ

доц., д-р Нора Ангелова

Шаблони на функции

- Шаблони на функции
  - Позволяват създаването на функции, използващи неопределени (хипотетични) типове данни за свои параметри или за резултата от обръщението към функцията.
- Чрез тях се описват „обобщени“ функции.

Шаблони на функции

- Да се напише функция, която въвежда елементите на масив от цели числа

```
void readIntArr(int* arr, int length) {
  for (int i = 0; i < length; i++) {
    std::cout << "arr[" << i << "]= ";
    std::cin >> arr[i];
  }
}
```

- Да се напише функция, която въвежда елементите на масив от реални числа

```
void readDoubleArr(double* arr, int length) {
  for (int i = 0; i < length; i++) {
    std::cout << "arr[" << i << "]= ";
    std::cin >> arr[i];
  }
}
```

Шаблони на функции

- Дефиниция на шаблон на функция

```
<шаблон_на_функция> ::=
template <<списък_от_параметри>>
<тип> <име_на_шаблон_на_функция> (<формални_параметри>) {
    <тяло>
},
```

където

```
<списък_от_параметри> ::=
typename <параметър> {= <тип>}опц
{, typename <параметър> {= <тип>}опц }опц
```

Пример:

```
template <typename T>
void readArr(T * arr, int length) {
  …
}
```

* Запазената дума typename е взаимозаменяема с class

Шаблони на функции

- Използване на шаблон на функция
  - Използването на дефинираните шаблони на функции се осъществява чрез обръщение към „обобщената“ функция, която шаблонът дефинира, но с параметри от конкретен тип.
  - Компилаторът генерира т. нар. шаблонна функция, като замества параметрите на шаблона с типовете на съответните фактически параметри.
  - При това заместване не се извършват преобразувания на типове.

ШАБЛОНИ НА ФУНКЦИИ

Пример:

Да се дефинира шаблон на функция за въвеждане на елементите на едномерен масив

```
template <typename T>
void readArrElements(T * arr, int length) {
  for (int i = 0; i < length; i++) {
    std::cout << "arr[" << i << "]= ";
    std::cin >> arr[i];
  }
}
…
const int ARR_LENGTH = 10;
int arr1[ARR_LENGTH];
double arr2[ARR_LENGTH];
readArrElements(arr1, ARR_LENGTH);
readArrElements(arr2, ARR_LENGTH);
```

* За T е дефиниран операторът >>

Шаблони на класове

- Шаблони на класове
  - Позволяват създаването на класове, използващи неопределени (хипотетични) типове данни за свои член-данни, за параметри на член-функции, за резултати от обръщения към член-функции.
  - Чрез тях се описват „обобщени“ класове – класове, зависещи от параметри.
  - Използват се за изграждане на общоцелеви класове - контейнери (динамични масиви, стекове, опашки, списъци и др.).

Шаблони на класове

- Декларация на шаблон на клас

<декларация_на_шаблон_на_клас> ::=

```
template <<списък_от_параметри>>
class <име_на_шаблон_на_клас> {
   // …
};
```

<списък_от_параметри> ::=

typename <параметър> {= <тип>}$_{опц}$
{, typename <параметър> {= <тип>}$_{опц}$ }$_{опц}$

Шаблони на класове

Пример:

```
template <typename T>
class TemplateClassName {
  // …
};
```

Шаблони на класове

- Подразбиращи се стойности

```
template <typename <параметър> = <подразбиращ_се_тип>
{, typename <параметър> = <подразбиращ_се_тип>}опц>
```

Пример:

```
template <typename T = int>
class TemplateClassName {
  …
};
```

Шаблони на класове

- Подразбиращи се стойности
  - Ако параметър е с подразбираща се стойност, всички параметри след него също трябва да са с подразбиращи се стойности.

Трябва да има
тип по подразбиране

Пример:

```
template <typename T = int, typename S = int>
class TemplateClassName {
  …
};
```

Шаблони на класове

- Дефинирането на член-функциите на шаблон на клас
  - Вградени (inline) член-функции
  - Извън тялото на класа

Шаблони на класове

Пример:

- Вградени член-функции

Пример:

```
template <typename T1, typename T2>
class TemplateClassName {
public:
  //…
  T1 inlineFunc(T1 x, T2 y) {  // Вградена член-функция
    //…
  }
  T2 testFunction(T1 x, T2 y);  // Член-функцията не е вградена
private:
  T1 firstField;
  T2 secondField;
};
```

ШАБЛОНИ НА КЛАСОВЕ

Пример:

- Извън тялото на класа
  - Дефиницията се предшества от template <<списък_от_параметри>>

Пример:

```
template <typename T1, typename T2>
T2 TemplateClassName<T1, T2>::testFunction(T1 x, T2 y) {
  //…
}
```

Шаблони на класове

- Тип може да бъде пропуснат – използва се типът по подразбиране, ако декларацията на шаблона е с подразбиращи се параметри, или се съобщава за грешка.
- Създаване на обект - компилаторът използва зададените (конкретни) типове и генерира съответен обект.

Пример:

```
TemplateClassName<int, double> obj;
```

Шаблони на класове

- Използване на `typedef` за задаване на ново име на специализация на шаблон на клас

Пример:

```
typedef TemplateClassName<int, double> NewNameClass;
```

Дефинира клас `NewNameClass` като специализация на шаблона на класа `TemplateClassName` при T - `int` и S - `double`.

Пример2:

```
typedef DynamicArr<int> IntArr;
```

Шаблони на класове

Забележка

Ако и двата параметъра на шаблона на класа `TemplateClassName` са със стойности по подразбиране, ще е възможно и дефиницията:

```
typedef TemplateClassName<> NewNameClass;
```

* Скобите <> трябва да присъстват.

Шаблони на клас

- Шаблонът на клас дефинира съвкупност от класове.
- Понякога се налага за конкретен тип данни член-функция на шаблона на класа да се реализира по по-различен алгоритъм.
- В C++ е възможно предефинирането на член-функция на шаблон на клас за конкретен тип.
  Този процес се нарича **специализация на член-функцията за съответния тип**.

Шаблони на клас

- Специализация на член-функция за съответния тип

```
class Point2D {
  double x;
  double y;
public:
  void print() const {
    std::cout << '(' << x << ',' << y << ')’;
  }
};
```

ШАБЛОНИ НА КЛАС

- Специализация на член-функция за съответния тип

```
template <typename T>
class TemplateClassName {
public:
  void print() const;
private:
  T value;
};

template <typename T>
void TemplateClassName<T>::print() const {
  std::cout << "template print function" << std::endl;
  std::cout << value;
}

template <>
void TemplateClassName<Point2D>::print() const {
  std::cout << "int print function" << std::endl;
  value.print();
}
```

Шаблони на клас

- (Не)типови параметри
  - Подобни на параметри на функции

Пример:

```
template <typename T, const int SIZE>
class StaticArr {
  T memoryBlock[SIZE];
  //...
};

int main() {
  StaticArr<int, 5> staticArrObj;
  return 0;
}
```

Шаблони на клас

- (Не)типови параметри със стойност по подразбиране

Пример:

```
template <typename T, const int SIZE=10>
class StaticArr {
  T memoryBlock[SIZE];
  //...
};

int main() {
  StaticArr<int> staticArrObj;
  return 0;
}
```

Шаблони на клас

- (Не)типови параметри от шаблонен тип

Пример:

```
template <typename T, const T PARAM>
class StaticArr {
  T memoryBlock[20];
  //...PARAM
};

int main() {
  StaticArr<int, 5> staticArrObj;
  return 0;
}
```

ВРЕМЕ ЗА ВАШИТЕ ВЪПРОСИ