Информатика. Семинар №7

05.04.2016

https://dl.dropboxusercontent.com/u/96739039/sem4/infa_s07.pdf

Шаблоны функций

- перегрузка функций = название одно и то же, но отличается тип или количество принимаемых параметров

```
int f(int x);
float f(int x); // error
```

P.S. https://en.wikipedia.org/wiki/Name_mangling

Шаблоны функций

```
int max(int a, int b)
{
  return a > b ? a : b;
}

float max(float a, float b)
{
  return a > b ? a : b;
}
```

Шаблоны функций

```
template <typename T>
T max(T a, T b)
{
  return a > b ? a : b;
}

int main()
{
  int c = max<int>(1.0f, 2);
  int d = max(1, 2);
  return 0;
}
```

Шаблоны функций

- Шаблонных параметров может быть несколько, могут задаваться типами по умолчанию, могут быть перечислимыми переменными (short, int, long):

```
template·<typename·T,·typename·U·=·T>
void·f(T·x,·U·y);

template·<typename·T,·int·size>
void·f(T·x)
{
··T·a[size];
··//·...
}
```

Шаблоны функций

- В качестве параметров в шаблоны можно передавать только значений известные на момент компиляции, т.е. написать так нельзя

```
template·<typename·T,·int·size>
void·f(T·x);

int·main()
{
··int·n;
··f<int,·n>(1);
··return·0;
}
```

Специализация шаблона

- А что если нашёлся какой-то тип, для которого не определен оператор <?

```
template·<>
const·MyInt&·max(const·MyInt&·a,·const·MyInt&·b)
{
··return·a.Compare(b)·?·a·:·b;
}
```

- Частичную специализацию шаблонов ф-й делать нельзя: нужно явно указывать сразу все шаблонные параметры

Шаблоны классов. Упражнение 1

Оценить асимптотику алгоритма сортировки

```cpp
#include <algorithm>
#include <iostream>

template <typename T>
struct Comparator {
  static size_t count;
  bool operator()(const T& x, const T& y) {
    ++count;
    return x < y;
  }
};

template<typename T>
size_t Comparator<T>::count = 0;

int main() {
  std::vector<int> v;
  // ... initialization
  std::sort(v.begin(), v.end(), Comparator<int>());
  return 0;
}
```

Шаблоны классов

```
template·<int·n>
struct·F
{
··enum·{
····result·=·n·*·F<n·-·1>::result
··};
};

template·<>
struct·F<0>
{
··static·const·int·result·=·1;
};

int·main()·{
··std::cout·<<·F<3>::result·<<·"·"·<<·F<4>::result;
··return·0;
}
```

Шаблоны классов. Упражнение 2

- На этапе компиляции вычислить C(n, k):

```
template·<int·n,·int·k>
struct·C
{
··//·TODO
};
```

Шаблоны классов

```
namespace std
{
  template <typename T, typename Allocator>
  class vector
  {
    //...
  };
}
```

```
std::vector<int> a;
std::vector<bool> b; // partial specialization
```

Частичная специализация шаблонов

```
template·<typename·T>
struct·IsPointer
{
··static·const·bool·result·=·false;
};

template·<typename·T>
struct·IsPointer<T*>
{
··static·const·bool·result·=·true;
};

int·main()·{
··std::cout·<<·IsPointer<float>::result·<<·"·"·<<·IsPointer<char*>::result;
··return·0;
}
```

Упражнение 3, 4

```
template·<typename·T>
class·Stack
{
public:
··T&·Top()·const;
··void·Pop();
··void·Push(const·T&·elem);
private:
··//·TODO
};

template·<typename·T,·int·size>
class·FixedSizeStack
{
··//·TODO
};
```