Информатика. Семинар №8

12.04.2016

https://dl.dropboxusercontent.com/u/96739039/sem4/infa_s08.pdf

Лямбда-функции

```
struct Comparator
{
  // TODO
};

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

Лямбда-функции

```
int count = 0;

auto comparator = [&](auto a, auto b) {
  count++;
  return a > b;
};

std::sort(v.begin(), v.end(), comparator);
```

1. Компилятор вместо auto сам «подставит» имя нужного типа.
2. Здесь comparator – функтор (класс, у которого определен оператор «круглые скобки» ())
3. [&] – значит, что ко всем переменным, переданным в качестве аргументов и используемым, обращение будет вестись по ссылке; [=] – по значению; [&a, =b] – первый аргумент по ссылке, второй – по значению.

Лямбда-функции

```
int count = 0;
std::sort(v.begin(), v.end(), [&](auto a, auto b)
    {
        count++;
        return a > b;
    });
```

Где ещё можно использовать лямбда-функции?

```
·std::for_each(v.begin(),·v.end(),·[&](int·x)·{
···std::cout·<<·x·<<·'·';
·});
```

```
int·count·=·std::count_if(v.begin(),·v.end(),·[](int·x)
··{
····return·x·<·5;
··});
```

Где ещё можно использовать лямбда-функции?

```cpp
// copy_if example
#include <iostream>      // std::cout
#include <algorithm>     // std::copy_if, std::distance
#include <vector>        // std::vector

int main () {
  std::vector<int> foo = {25,15,5,-5,-15};
  std::vector<int> bar (foo.size());

  // copy only positive numbers:
  auto it = std::copy_if (foo.begin(), foo.end(), bar.begin(), [](int i){return !(i<0);} );
  bar.resize(std::distance(bar.begin(),it));  // shrink container to new size

  std::cout << "bar contains:";
  for (int& x: bar) std::cout << ' ' << x;
  std::cout << '\n';

  return 0;
}
```

http://www.cplusplus.com/reference/algorithm/ - если можно воспользоваться какой-то готовой функцией из стандартной библиотеки, то лучше так и сделать, а не писать свою (ещё одну) реализацию.

Упражнение 1

Дано число N и далее N строк. Найти количество уникальных *эквивалентных* строк и вывести их.

P.S. Две строки назовём *эквивалентными*, если после удаления пробелов и перевода символов в нижний регистр они совпадают.

P.P.S. Размер программы должен быть как можно меньше - активно используйте лямбда-функции. Обратите внимание на ф-и
http://www.cplusplus.com/reference/algorithm/transform/,
std::sort,
http://www.cplusplus.com/reference/algorithm/unique/
Для работы со строками используйте класс std::string.

Потоки в C++11

```cpp
#include <iostream>
#include <thread>

void func0() {
  std::cout << "Hi! I'm function 0\n";
}

void func1(int x) {
  std::cout << "Hi! I'm function 1 " << x << "\n";
}

int main() {
  std::thread t0(func0);
  std::thread t1(func1, 42);

  t0.join();
  t1.join();
```

Потоки + лямбда-функции

```cpp
#include <iostream>
#include <thread>

int main() {
  std::thread t0([] () {
    std::cout << "Hi! I'm function 0\n";
  });

  std::thread t1([] (int x) {
    std::cout << "Hi! I'm function 1 " << x << "\n";
  }, 42);

  t0.join();
  t1.join();
```

Потоки в C++11

```
#include·<future>·
#include·<numeric>·//·std::accumulate
#include·<functional>·//·std::plus

const·int·N·=·8;·const·int·Size·=·1000000;

int·main()·{
··std::vector<int>·v(N·*·Size);
··for·(int&·x·:·v)·x·=·rand()·%·10;

··std::chrono::high_resolution_clock::time_point·start·=·std::chrono::high_resolution_clock::now();

··std::vector<·std::future<int>·>·f;
··for·(int·i·=·0;·i·<·N;·++i)·{
····f.push_back(std::async([&v,·i]()·{
··········int·sum·=·std::accumulate(v.begin()·+·i·*·Size,·v.begin()·+·(i·+·1)·*·Size,·
···········0,·std::plus<int>());
··········return·sum;
·····});
····);
··}

··for·(auto&·result·:·f)·std::cout·<<·result.get()·<<·"·";
··std::cout·<<·std::endl;

··std::chrono::duration<double>·diff·=·std::chrono::high_resolution_clock::now()·-·start;
··std::cout·<<·diff.count()·<<·"s";
```

Аналог progress bar`a

```
·std::chrono::milliseconds·span(10);
·while·(f.back().wait_for(span)·==·std::future_status::timeout)
···std::cout·<<·"."·<<·std::flush;
```

Упражнение 2

Методом Монте-Карло посчитать объём сферы в несколько потоков + измерить ускорение.

P.S. Можно и n-мерной: https://en.wikipedia.org/wiki/N-sphere