Elementy STLa

Czyli jak nie odkrywać koła na nowo.

Co to jest STL?

- **Standard Template Library**, **STL** (ang. standardowa biblioteka szablonów)- biblioteka C++ zawierająca algorytmy, pojemniki, iteratory oraz inne konstrukcje w formie szablonów, gotowe do użycia w programach.

Elementy STL: pojemniki

- Pojemnik = obiekt zbiorczy
- Vector: taka tablica dynamiczna o swobodnym dostępie. Słaba wydajność czasowa wstawiania (bo elementy trzeba „przesuwać”).
- List: lubiana przez Was lista jednokierunkowa ☺
- Queue, Stack – znane z algorytmów ☺
- Map, Multimap – tzw. tablice asocjacyjne, pomocne np. do budowania słowników.

Vector

- Deklaracja:
  - vector<int> tab; //nie podajemy rozmiaru!
- Przydatne metody:
  - tab.push_back(2); - wstawia element na koniec
  - tab.insert(tabliczka.begin(), 5); - wstawia na początek
  - tab.size() – zwraca aktualny rozmiar
  - tab.clear() – czyści wektor
  - tab[10] – odwołanie do 10 elementu

List, queue, stack

- Listy:
  - http://www.sgi.com/tech/stl/List.html
- Kolejki:
  - http://www.sgi.com/tech/stl/queue.html
- Stosy:
  - http://www.sgi.com/tech/stl/stack.html

Całość:

- http://www.yolinux.com/TUTORIALS/LinuxTutorialC++STL.html
- http://pl.wikibooks.org/wiki/C%2B%2B#Dodatek_A:_Biblioteka_STL

Map, multimap

```
map<string, int> slownik;
while(file.hasNextString()) //czytamy z pliku słowa
        {
        string slowo <- nowe słowo z pliku;
        if(slownik.count(slowo)) slownik[slowo]++;
            else slownik.insert(pair<string,int>("Ala",1));
        }
```

Ćwiczenia

1. Wczytaj plik tekstowy, rozbij go na poszczególne litery i przedstaw analizę częstości występowania tychże (map)
2. Napisz program do generowania losowych liczb korzystając z vectora. Użytkownik podaje ile tych liczb ma być. Posortuj je malejąco.
3. Zadeklaruj dwa stosy w programie. Za pomocą drugiego stosu, odwróć elementy ze stosu pierwszego

Iteratory

- Ułatwiają pracę na kontenerach
- Brak konieczności znajomości struktury (inaczej: tak samo działają dla vectora, listy, stosu, tablicy, itp.)
- Podobne w idei do wskaźników, ale mniej błędogenne ☺

http://pl.wikibooks.org/wiki/C%2B%2B/Iteratory

Iteratory - przykład

```
5   using namespace std;
6
7   int main()
8   {
9       srand(time(NULL));
10      list<int> lista;
11      list<int>::iterator itW=lista.begin();
12      for(int i=0;i<50;++i)
13          lista.insert(rand()%100);
14      for(itW=lista.begin();itW!=lista.end();itW++) {
15      cout << (*itW) << endl;
16      }
17
18      list<int>::iterator it;
19      it = lista.end();
20      lista.insert(it, 1);
21      lista.insert(it,2);
22      lista.insert(--it,3);
23      return 0;
24  }
25
```

Algorytmy

- Dodajemy nagłówek:
  - #include <algorithm>
- Podstawowe operacje na kontenerach zostały już zaimplementowane. I tak do dyspozycji mamy na przykład:
  - sortowanie (sort())
  - wyszukiwanie(find(), find_first())
  - wypełnianie (copy(), fill(), replace())
  - zmiana kolejności (swap(), replace(), random_shuffle())

i wiele innych:
http://pl.wikibooks.org/wiki/C%2B%2B/Algorytmy_w_STL