Politechnika Świętokrzyska

Laboratorium

Programowanie w języku C++ 2

Ćwiczenie 4

Podstawowe metody kontenerów

Cel ćwiczenia

Celem ćwiczenia jest zapoznanie studentów z biblioteką STL języka C++.

dr inż Robert Kazała

size() i max_size()

Przykład 1

```
#include <list>
#include <vector>
#include <deque>
#include <iostream>

using namespace std;

int main()
{
    int a[] = {1,2,3,4,5,6,7,8,9,10};
    //containers
    vector <int> v(a,a+10);
    deque <int> d(a,a+10);
    list <int> l(a,a+10);
    cout<<"Size of vector, deque, list: "<<v.size() << " " << d.size()
<<" "
        << l.size()<<endl;
    cout<<"Max size of vector, deque, list: "<<v.max_size() << " "
        << d.max_size() <<" " << l.max_size()<<endl<<endl;

    v.push_back(11);
    d.push_back(11);
    l.push_back(11);

    cout<<"Size of vector, deque, list: "<<v.size() << " " << d.size()
<<" "
        << l.size()<<endl;
    cout<<"Max size of vector, deque, list: "<<v.max_size() << " "
        << d.max_size() <<" " << l.max_size()<<endl<<endl;

    v.pop_back();
    d.pop_back();
    l.pop_back();

    cout<<"Size of vector, deque, list: "<<v.size() << " " << d.size()
<<" "
        << l.size()<<endl;
    cout<<"Max size of vector, deque, list: "<<v.max_size() << " "
        << d.max_size() <<" " << l.max_size()<<endl<<endl;
    return 0;
}
```

empty() i resize()

Przykład 2

```
#include <list>
#include <vector>
#include <deque>
#include <iostream>

using namespace std;

int main()
{
    //containers
    vector <int> v;
    deque <int> d;
    list <int> l;
    cout<<"Size of vector, deque, list: "<<v.size() << " " << d.size()
<<" "
        << l.size()<<endl;

    if (v.empty())
    {
        v.resize(10);
    }
    if (d.empty())
    {
        d.resize(10);
    }
    if (l.empty())
    {
        l.resize(10);
    }
    cout<<"Size of vector, deque, list: "<<v.size() << " " << d.size()
<<" "
        << l.size()<<endl;

    if (!v.empty())
    {
        v.resize(9);
    }
    if (!d.empty())
    {
        d.resize(5);
    }
    if (!l.empty())
    {
        l.resize(0);
    }
    cout<<"Size of vector, deque, list: "<<v.size() << " " << d.size()
<<" "
        << l.size()<<endl;
    return 0;
}
```

vector::capacity() – tylko vector

Przykład 3

```
#include <vector>
#include <iostream>

using namespace std;

int main()
{
    vector <int> v;
    cout<<"Size and capacity: "<<v.size() << " " << v.capacity() <<endl;

    for(int i = 0; i < 20; ++i)
    {
        v.push_back(i);
        cout<<"Size and capacity: "<<v.size() << " " << v.capacity()
<<endl;
    }
    return 0;
}
```

vector::reserve() – tylko vector

Przykład 4

Generowanie wektora liczb losowych

```
#include <vector>
#include <iostream>

using namespace std;

int main()
{
    vector <int> v;
    cout<<"Size and capacity: "<<v.size() << " " << v.capacity() <<endl;
    v.reserve(15);
    cout<<"Size and capacity: "<<v.size() << " " << v.capacity() <<endl;
    cout<<"Adding elements"<<endl;
    for(int i = 0; i < 10; ++i)
    {
        v.push_back(i);
        cout<<"Size and capacity: "<<v.size() << " " << v.capacity()
<<endl;
    }
    cout<<"Trying to shrink..."<<endl;
    v.reserve(10);
    cout<<"Size and capacity: "<<v.size() << " " << v.capacity() <<endl;
    return 0;
}
```

back() i front()

Przykład 5

```
#include <list>
#include <vector>
#include <deque>
#include <iostream>

using namespace std;

template<class I>
void print (const I & start, const I & end)
{
    I it;
    for(it = start; it != end; ++it)
    {
        cout<< *it << " ";
    }
    cout<<endl;
}

int main()
{
    int a[] = {1,2,3,4,5,6,7,8,9,10};
    //containers
    vector <int>  v(a,a+10);
    deque <int>  d(a,a+10);
    list <int>  l(a,a+10);

    cout<<"Size of vector, deque, list: "<<v.size() << " " << d.size()
        <<" " << l.size()<<endl;
    cout<<"Values at front (vector, deque, list): "<< v.front() << " "
<<d.front()
        << " " <<l.front() <<endl;
    cout<<"Size of vector, deque, list: "<<v.size() << " " << d.size()
        <<" " << l.size()<<endl;

    v.front() = 100;
    d.front() = 101;
    l.front() = 102;
    print(v.begin(), v.end());
    print(d.begin(), d.end());
    print(l.begin(), l.end());
    return 0;
}
```

Przykład 6

```
#include <list>
#include <vector>
#include <deque>
#include <iostream>

using namespace std;
```

```
template<class I>
void print (const I & start, const I & end)
{
    I it;
    for(it = start; it != end; ++it)
    {
        cout<< *it << " ";
    }
    cout<<endl;
}

int main()
{
    int a[] = {1,2,3,4,5,6,7,8,9,10};
    //containers
    vector <int>  v(a,a+10);
    deque <int>  d(a,a+10);
    list <int> l(a,a+10);

    cout<<"Size of vector, deque, list: "<<v.size() << " " << d.size()
        <<" " << l.size()<<endl;
    cout<<"Values at back (vector, deque, list): "<< v.back() << " "
<<d.back()
        << " " <<l.back() <<endl;
    cout<<"Size of vector, deque, list: "<<v.size() << " " << d.size()
        <<" " << l.size()<<endl;

    v.back() = 100;
    d.back() = 101;
    l.back() = 102;
    print(v.begin(), v.end());
    print(d.begin(), d.end());
    print(l.begin(), l.end());
    return 0;
}
```

operator[] i at() – tylko vector i deque

Przykład 7

```
#include <vector>
#include <deque>
#include <iostream>

using namespace std;

template <class C>
void print (const C & container)
{
    for(unsigned i = 0; i < container.size(); ++i)
    {
        cout<< container[i] << " ";
    }
    cout<<endl;
}
```

```
int main()
{
    int a[] = {1,2,3,4,5,6,7,8,9,10};
    //containers
    vector <int> v(10);
    deque <int> d(10);

    for(unsigned i = 0; i < 10; ++i)
    {
        d[i] = a[i];
        v[i] = a[i];
    }
    print(v);
    print(d);
    cout<<"Accessing out of range element:\n";
    cout<<v[10]<<" " <<d[10]<<endl;

    return 0;
}
```

Przykład 8

```
#include <vector>
#include <deque>
#include <iostream>

using namespace std;

template <class C>
void print (const C & container)
{
    for(unsigned i = 0; i < container.size(); ++i)
    {
        cout<< container.at(i) << " ";
    }
    cout<<endl;
}

int main()
{
    int a[] = {1,2,3,4,5,6,7,8,9,10};
    //containers
    vector <int>  v(10);
    deque <int>  d(10);

    for(unsigned i = 0; i < 10; ++i)
    {
        d.at(i) = a[i];
        v.at(i) = a[i];
    }
    print(v);
    print(d);
    cout<<"Accessing out of range element:\n";
    try
    {
```

```
        cout<<v.at(11)<<endl;
    }
    catch (out_of_range & ex)
    {
        cout<< ex.what()<<endl;
    }

    try
    {
        cout<<d.at(11)<<endl;
    }
    catch (out_of_range & ex)
    {
        cout<< ex.what()<<endl;
    }


    return 0;
}
```

assign()

Przykład 9

```
#include <list>
#include <vector>
#include <deque>
#include <iostream>

using namespace std;

template<class I>
void print (const I & start, const I & end)
{
    I it;
    for(it = start; it != end; ++it)
    {
        cout<< *it << " ";
    }
    cout<<endl;
}

int main()
{
    int a[] = {1,2,3,4,5,6,7,8,9,10};
    //containers
    vector <int> v(a,a+5);
    deque <int> d(a,a+5);
    list <int> l(a,a+5);

    print(v.begin(), v.end());
    print(d.begin(), d.end());
    print(l.begin(), l.end());

    cout<<"Assigning a new content:\n";
```

```
    v.assign(a, a+10);
    d.assign(a, a+10);
    l.assign(a, a+10);
    print(v.begin(), v.end());
    print(d.begin(), d.end());
    print(l.begin(), l.end());

    cout<<"Assigning a new content II:\n";
    v.assign(3, 100);
    d.assign(3, 1000);
    l.assign(3, 10000);
    print(v.begin(), v.end());
    print(d.begin(), d.end());
    print(l.begin(), l.end());
    return 0;
}
```

insert()

Przykład 10

```
#include <vector>
#include <iostream>

using namespace std;

template <class I>
void print (const I & start, const I & end)
{
    I it;
    for(it = start; it != end; ++it)
    {
        cout<< *it << " ";
    }
    cout<<endl;
}

int main()
{
    int a[] = {1,2,3,4,5,6,7,8,9,10};
    vector <int> v(a,a+10);

    vector<int>::iterator it = v.insert(v.begin()+5, 100);
    print(v.begin(), v.end());
    cout<<"Inserted element: "<<*it<<endl;
    cout <<"Size: "<<v.size()<<endl;

    vector <int> v2;
    v2.insert(v2.begin(), v.rbegin(), v.rend());
    print(v2.begin(), v2.end());

    vector <int> v3(v.begin(), v.begin()+5);
    v3.insert(v3.end(),3,100);
```

```
    print(v3.begin(), v3.end());

    return 0;
}
```

Przykład 11

```
#include <deque>
#include <iostream>

using namespace std;

template<class I>
void print (const I & start, const I & end)
{
    I it;
    for(it = start; it != end; ++it)
    {
        cout<< *it << " ";
    }
    cout<<endl;
}

int main()
{
    int a[] = {1,2,3,4,5,6,7,8,9,10};
    deque <int> d1(a,a+10);

    deque<int>::iterator it = d1.insert(d1.begin()+5, 100);
    print(d1.begin(), d1.end());
    cout<<"Inserted element: "<<*it<<endl;
    cout <<"Size: "<<d1.size()<<endl;

    deque <int> d2;
    d2.insert(d2.begin(), d1.rbegin(), d1.rend());
    print(d2.begin(), d2.end());

    deque <int> d3(d1.begin(), d1.begin()+5);
    d3.insert(d3.end(),3,100);

    print(d3.begin(), d3.end());

    return 0;
}
```

Przykład 12

```
#include <list>
#include <iostream>

using namespace std;

template<class I>
```

```
void print (const I & start, const I & end)
{
    I it;
    for(it = start; it != end; ++it)
    {
        cout<< *it << " ";
    }
    cout<<endl;
}

int main()
{
    int a[] = {1,2,3,4,5,6,7,8,9,10};
    list <int> l1(a,a+10);

    list<int>::iterator it = l1.insert(l1.begin(), 100);
    print(l1.begin(), l1.end());
    cout<<"Inserted element: "<<*it<<endl;
    cout <<"Size: "<<l1.size()<<endl;

    list <int> l2;
    l2.insert(l2.begin(), l1.rbegin(), l1.rend());
    print(l2.begin(), l2.end());

    list<int>::iterator it1 = l1.begin();
    //shifting 5 places
    for(int i = 0; i < 5; ++i)
    {
        ++it1;
    }
    list <int> l3(l1.begin(), it1);
    l3.insert(l3.end(),3,100);

    print(l3.begin(), l3.end());

    return 0;
}
```

erase()

Przykład 13

```
#include <list>
#include <vector>
#include <deque>
#include <iostream>

using namespace std;

template<class I>
void print (const I & start, const I & end)
{
    I it;
    for(it = start; it != end; ++it)
    {
```

```
        cout<< *it << " ";
    }
    cout<<endl;
}

int main()
{
    int a[] = {1,2,3,4,5,6,7,8,9,10};
    vector <int> v(a,a+10);
    deque <int> d(a,a+10);
    list <int> l(a,a+10);

    print(v.begin(), v.end());
    print(d.begin(), d.end());
    print(l.begin(), l.end());

    cout<<"Ereasing elements:\n";
    v.erase(v.begin()+3);
    d.erase(d.begin()+3);
    //no random access iterator
    list<int>::iterator it= l.begin();
    ++it; ++it; ++it;
    it = l.erase(it);
    print(v.begin(), v.end());
    print(d.begin(), d.end());
    print(l.begin(), l.end());

    cout<<"Ereasing elements:\n";
    v.erase(v.begin()+3, v.end());
    d.erase(d.begin()+3, d.end());
    l.erase(it, l.end());
    print(v.begin(), v.end());
    print(d.begin(), d.end());
    print(l.begin(), l.end());
    return 0;
}
```

Literatura

Zadania

1. Przeanalizować, uruchomić, zmodyfikować i opisać kod wszystkich przykładów z instrukcji.