

Trabajo En Clase

Standard Template Library

Estructura de Datos 2023 - 30

Pontificia Universidad Javeriana

Mariana Diaz Puentes

9/08/2023

STL00:

```
#include <iostream>
#include <stack>
using namespace std;

int main() {
    stack<int> st;
    st.push(10);
    st.push(20);
    st.push(30);
    st.push(40);

    st.pop();
    st.pop();

    while (!st.empty())
    {
        cout<< " " << st.top();
        st.pop();
    }
    cout << "\n....fin \n";
}
```

Este código quiere que el programa agregue cuatro stacks a una pila, elimine los dos elementos superiores y luego imprima los elementos restantes de la pila. El cual se vería así, usando los siguientes comandos en la shell.

```
~/ayuda$ g++ -std=c++11 -g -o m0 stl00.cpp
~/ayuda$ ./m0
20 10
....fin
```

STL01

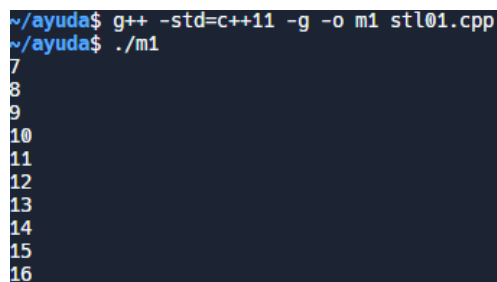
```
#include <iostream>
using namespace std;

template<class T, int element> class Exp{
public:
    T a [ element];
    void put (){
        int x = 1;
        for (int j=0; j<element;j++)
        {
            a[j] =x;
            x++;
        }
    }
    void print (){
        for (int x=6; x<element;x++)
            cout<< a[x]<<" "<<"\n";
        cout << "\n.....FIN! \n";
    }
};

int main(){
    Exp<int,16> objeto;
    objeto.put();
    objeto.print();
    return 0;
}
```

El código muestra cómo usar una plantilla de clase para hacer un objeto con un array de tamaño definido y hacer operaciones simples como inicializar y mostrar los valores almacenados en el array.

Una nota importante es que en el método print, hay un error en el bucle for. Debería comenzar desde el índice 6 en lugar de 0, ya que el array se inicializa con valores desde 1, y luego se intenta imprimir los valores desde el índice 6.



```
~/ayuda$ g++ -std=c++11 -g -o m1 stl01.cpp
~/ayuda$ ./m1
7
8
9
10
11
12
13
14
15
16
.....FIN!
```

STL02

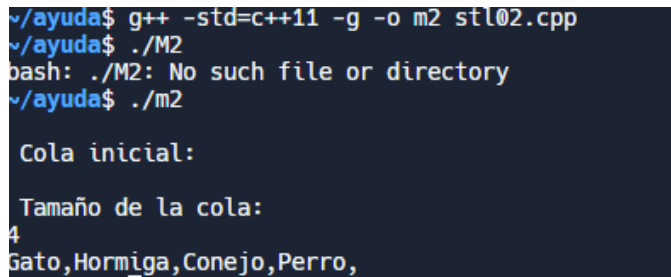
```
#include <bits/stdc++.h>
using namespace std;
void toolCola(queue <string> objeto);

int main(){
    // create a queue of string
    queue <string> animales;
    animales.push("Gato");
    animales.push("Hormiga");
    animales.push("Conejo");
    animales.push("Perro");

    cout << "\n Cola inicial: \n";
    toolCola(animales);
    return 0;
}

void toolCola(queue <string> objeto)
{
    cout << "\n Tamaño de la cola: \n" << objeto.size() << "\n";
    while (!objeto.empty())
    {
        cout << objeto.front() << ", ";
        objeto.pop();
    }
    cout << endl;
}
```

En resumen, el código crea una cola de cadenas llamada animales, agrega algunas cadenas a la cola y luego utiliza una función toolCola para imprimir el tamaño de la cola y los elementos almacenados en ella. La función toolCola recorre la cola, imprime los elementos uno por uno y finalmente imprime un salto de línea.



```
~/ayuda$ g++ -std=c++11 -g -o m2 stl02.cpp
~/ayuda$ ./M2
bash: ./M2: No such file or directory
~/ayuda$ ./m2

Cola inicial:

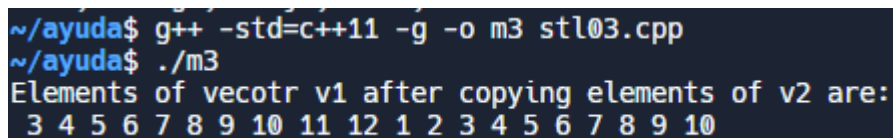
Tamaño de la cola:
4
Gato,Hormiga,Conejo,Perro,
```

STL03

```
#include <iostream>
#include <queue>
using namespace std;

int main (){
    vector<int> v1, v2;
    for(int i=1;i<=10; i++){
        v1.push_back(i);
        v2.push_back(i+2);
    }
    vector<int>::iterator itr = v1.begin();
    copy(v2.begin(),v2.end(), inserter(v1,itr));
    cout<<"Elements of vecotr v1 after copying elements of v2 are: "<<endl;
    for(itr=v1.begin(); itr!= v1.end();++itr){
        cout<<" "<< *itr;
    }
    cout <<"\n\n";
    return 0;
}
```

En resumen, el código crea dos vectores (v1 y v2), llena v1 con números del 1 al 10 y v2 con números del 3 al 12. Luego, utiliza el concepto de iteradores y la función copy para insertar los elementos de v2 en v1. Finalmente, imprime los elementos de v1 después de realizar la copia.



```
~/ayuda$ g++ -std=c++11 -g -o m3 stl03.cpp
~/ayuda$ ./m3
Elements of vecotr v1 after copying elements of v2 are:
 3 4 5 6 7 8 9 10 11 12 1 2 3 4 5 6 7 8 9 10
```

STL04

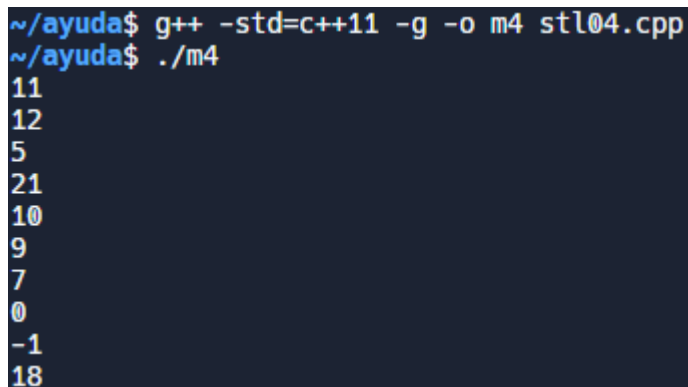
El programa es

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

using namespace std;

int main() {
    list<int> lista00={12,5,10,9,7,0,-1};
    lista00.push_front(11);
    lista00.push_back(18);
    auto it= std::find(lista00.begin(), lista00.end(),10);
    if(it!= lista00.end()){
        lista00.insert(it,21);
    }
    for(int x: lista00){
        cout << x << '\n';
    }
    return 0;
}
```

En resumen, el código crea una lista de enteros, realiza diversas operaciones de inserción y búsqueda en la lista, e imprime los elementos resultantes después de todas las operaciones.



```
~/ayuda$ g++ -std=c++11 -g -o m4 stl04.cpp
~/ayuda$ ./m4
11
12
5
21
10
9
7
0
-1
18
```

STL 05

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

using namespace std;

int main(void){
    list<int> lista00;
    list<int> lista01= {10,20,30};
    list<int> lista02(lista01.begin(), lista01.end());
    list<int> lista03(move(lista01));
    cout << "Tamaño de Lista00"<<lista00.size() << endl;
    cout << "Contenido de Lista02"<< endl;

    for(auto it= lista02.begin(); it !=lista02.end();it++)
        cout<< *it <<endl;

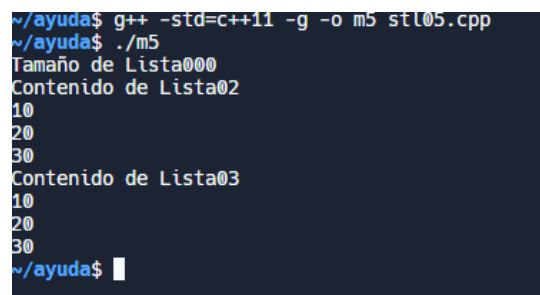
    cout << "Contenido de Lista03"<< endl;

    for(auto it= lista03.begin(); it !=lista03.end();it++)
        cout<< *it <<endl;

    return 0;

}
```

En resumen, el código crea y maneja listas de enteros utilizando la biblioteca list de C++. Muestra cómo inicializar listas con diferentes enfoques, cómo imprimir sus tamaños y cómo recorrer e imprimir los elementos de las listas utilizando bucles for.



```
~/ayuda$ g++ -std=c++11 -g -o m5 stl05.cpp
~/ayuda$ ./m5
Tamaño de Lista00
Contenido de Lista02
10
20
30
Contenido de Lista03
10
20
30
~/ayuda$
```

STL 06

```
#include <iostream>
#include <set>
using namespace std;

int main()
{
    set<string> conjunto00{"iphone", "andorid", "basic", "landline"};
    set<int> conjunto02{1,2,3,4,5};
    set<char> conjunto01 {'a', 'b', 'c', 'd'};
    int i=5;

    for (auto it =conjunto00.begin(); it !=conjunto00.end(); ++it,++i)
    {
        conjunto02.insert('a'+i);
    }

    cout <<"Tamño del conjunto:" << conjunto00.size();
    cout<<endl;
    cout <<"Tamño del conjunto:" << conjunto01.size();
    cout<<"\n";
    return 0;
}
```

En resumen, el código crea conjuntos de diferentes tipos (string, int y char), inicializa los conjuntos con valores y realiza una operación inusual de inserción en el conjunto conjunto02. Luego, imprime los tamaños de conjunto00 y conjunto01 en la salida estándar. Además, que no permite que se repita ninguna parte del conjunto