Приклад використання шаблонного типу даних

template<typename T1> //Один раз описали і можемо використовувати у цілому класі

class MyClass

{

public:

T1 Var1; //можемо використовуват цей тип для опису полів і методів

MyClass(T1 var1)

{

Var1 = var1;

}

T1 multiplyBy2()

{

return Var1 \* 2;

}

};

//--------------------

//Якщо вказуємо в якості предка, то тип даних потрібно зазначити

class Deriv1 :public MyClass<double>

{

};

//-----------------------

//Якщо нащадок також містить узагальнений тип даних, то його можна передати у предок

template<typename T2>

class Deriv2 :public MyClass<T2>

{

};

int main()

{

//При створенні об’єкта потрібно чітко вказати, чим є шаблонний тип даних

MyClass<int> cl1(12);

cout << cl1.multiplyBy2()<<endl;

MyClass<double>\* p = new MyClass<double>(3.5);

cout << p->multiplyBy2() << endl;

MyClass<char> cl2('A');

cout << cl2.multiplyBy2() << endl;

MyClass<string> cl3("hello");

cout << cl3.multiplyBy2().data() << endl;

system("pause");

return 0;

}

//---------------

**Задача. Створити клас, який представляє масив і дозволяє зберегти дані довільного типу**

//-------------------- Array.h -----------------------

#pragma once

#include<string>

#include<fstream>

using namespace std;

template<typename T>

class Array

{

T\* data;

int length;

int maxLength;

void setMaxLength(int maxLength)

{

data = new T[maxLength];

this->maxLength = maxLength;

length = 0;

}

public:

int getLength()

{

return length;

}

int getMaxLength()

{

return maxLength;

}

void add(T value)

{

if (length==maxLength)

{

T\*newData = new T[maxLength \* 2];

for (int i = 0; i < length; i++)

{

newData[i] = data[i];

}

delete[] data;

data = newData;

maxLength \*= 2;

}

data[length] = value;

length++;

}

int indexOf(T value)

{

for (int i = 0; i < length; i++)

{

if (data[i]==value)

{

return i;

}

}

return -1;

}

void remove(T value)

{

int valueIndex = indexOf(value);

if (valueIndex!=-1)

{

removeAt(valueIndex);

}

}

void removeAt(int position)

{

if (position>=0 && position<length)

{

for (int i = position+1; i < length; i++)

{

data[i - 1] = data[i];

}

length--;

}

else

{

throw "position out of range";

}

}

string toString()

{

string s = to\_string(data[0]);

for (int i = 1; i < length; i++)

{

s += ", " + to\_string(data[i]);

}

return s;

}

void saveToFile(string fileName)

{

ofstream file(fileName);

file << length << endl;

for (int i = 0; i < length; i++)

{

file << data[i] << " ";

}

file.close();

}

void loadFromFile(string fileName)

{

length = 0;

ifstream file(fileName);

int temp;

file >> temp;

for (int i = 0; i < temp; i++)

{

T value;

file >>value;

add(value);

}

file.close();

}

//------------------------

Array(int maxLength)

{

setMaxLength(maxLength);

}

Array(string fileName) :Array()

{

loadFromFile(fileName);

}

Array():Array(20)

{}

~Array()

{

delete[] data;

}

};

..=================================== Main

// ConsoleApplication105.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include <iostream>

#include <string>

#include <fstream>

#include "Array.h"

using namespace std;

int main()

{

Array<int> arr;

arr.add(2);

arr.add(3);

arr.add(4);

arr.add(5);

arr.add(6);

arr.remove(4);

arr.add(77);

arr.saveToFile("My.txt");

Array<int> arr\_("My.txt");

cout << arr\_.toString().data() << endl;

system("pause");

cout << arr.toString().data() << endl;

Array<double> arr2;

arr2.add(12.8);

arr2.add(89.34);

arr2.add(21);

arr2.add(90.5);

arr2.add(11);

cout << arr2.toString().data() << endl;

//Array<string> arr3;

//arr3.add("Ivan");

//arr3.add("Petro");

//arr3.add("Stepan");

////cout << arr3.toString().data() << endl;

system("pause");

return 0;

}