**Лабораторна робота №2**

З дисципліни «ООП»

«Розробка сімейства поліморфних класів»

Виконав: Землянський Едуард

Група: КВ-22

**Постановка задачі**

A close up of a document

Description automatically generated

**Варіант завдання**

**A white background with black text

Description automatically generated**

**A white paper with black text

Description automatically generated**

**Вимоги до програми**

**Текст програми**

*Containers.hpp*

#ifndef Containers\_hpp

#define Containers\_hpp

#include <stdio.h>

#include <stdarg.h>

#include <string>

#include <iostream>

#include <typeinfo>

using namespace std;

// class BaseContainer{

// public:

//     BaseContainer(){}

//     ~BaseContainer(){}

//     virtual void show\_data();

//     virtual void get\_data\_type();

//     virtual void set\_data();

// };

// class BaseArrayContainer: public BaseContainer{

// public:

//     BaseArrayContainer(){}

//     ~BaseArrayContainer(){}

//     virtual void resize(int newSize);

// };

template <typename TYPE>

class Container{

private:

    TYPE data;

public:

    Container(){}

    Container(TYPE input) {

        data = input;

    }

    void set\_data(TYPE input) {

        data = input;

    }

    void set\_data(){

        cout << "Input data: ";

        cin >> data;

    }

    void show\_data()  {

        cout << "Stored data: " << data << endl;

    }

    TYPE get\_data() {

        return data;

    }

    void copy(const Container <TYPE> Container){

        data = Container.data;

    }

    void get\_data\_type()  {

        char type = typeid(data).name()[0];

        string type\_name;

        switch (type) {

        case 'i':

            type\_name = "int";

            break;

        case 'd':

            type\_name = "double";

            break;

        case 'c':

            type\_name = "char";

            break;

        case 'b':

            type\_name = "bool";

            break;

        case 'f':

            type\_name = "float";

            break;

        default:

            type\_name = "undefined";

            break;

        };

        cout << "Type: " << type\_name << endl;

    }

};

template <typename TYPE>

class ArrayContainer{

private:

    TYPE \* data;

    int size;

public:

    ArrayContainer(){}

    ArrayContainer(int input\_size, TYPE \* input\_data) : size(input\_size), data(input\_data){}

    ArrayContainer(int input\_size, TYPE input\_data) : size(input\_size){

        data = new TYPE[size];

        for (int index = 0; index < size; index++){

            data[index] = input\_data;

        }

    }

    ArrayContainer(int input\_size, ...) {

        size = input\_size;

        data = new TYPE[size];

        va\_list arguments;

        va\_start(arguments, input\_size);

        for (size\_t index = 0; index < size; index++) {

            if (typeid(TYPE) == typeid(char)) {

                data[index] = static\_cast <TYPE> (va\_arg(arguments, int));

            } else if (typeid(TYPE) == typeid(bool)) {

                data[index] = static\_cast <TYPE> (va\_arg(arguments, int));

            }

            else{

                data[index] = va\_arg(arguments, TYPE);

            }

        }

        va\_end(arguments);

    }

    ~ArrayContainer(){

        delete[] data;

    }

    Container <TYPE> at(int index){

        Container <TYPE> returned(data[index]);

        return returned;

    }

    void check\_size(){

        cout << "Array size: " << size << endl;

    }

    void show\_data() {

        for (int i = 0; i < size; i++) {

            if (typeid(TYPE) == typeid(bool)){

                if (data[i] == 1) cout << "true ";

                else cout << "false ";

                continue;

            }

            cout << data[i] << " ";

        }

        cout << endl;

    }

    void get\_data\_type(){

        Container <TYPE> returned(data[0]);

        returned.get\_data\_type();

    }

    void copy(const ArrayContainer <TYPE> Array){

        size = Array.size;

        data = Array.data;

    }

    void resize(int newSize) {

    TYPE \*newData = new TYPE[newSize];

    for (int index = 0; index < newSize; index++) {

        if (index >= size) {

            if (typeid(TYPE) == typeid(char)) {

                newData[index] = '\0';

            } else {

                newData[index] = 0;

            }

        } else {

            newData[index] = data[index];

        }

    }

    data = newData;

    size = newSize;

    }

};

#endif

**Тестування програми**

*Main.cpp*

#include <iostream>

#include <string>

#include "Containers.hpp"

using namespace std;

int main(){

    Container <int> number, number2;

    Container <char> character('a');

    number.set\_data(3);

    number2.set\_data(6);

    number2.copy(number);

    number2.show\_data();

    number.show\_data();

    number.set\_data();

    number.show\_data();

    character.show\_data();

    character.set\_data();

    character.show\_data();

    character.get\_data\_type();

    number.get\_data\_type();

    Container <float> f(1.98);

    Container <double> d(1.1);

    Container <bool> b(1);

    b.show\_data();

    b.get\_data\_type();

    f.show\_data();

    d.show\_data();

    f.get\_data\_type();

    d.get\_data\_type();

    int inp\_d[] = {1, 2, 3, 4, 5};

    ArrayContainer <int> int\_arr(5, inp\_d);

    int\_arr.at(3).show\_data();

    int\_arr.show\_data();

    ArrayContainer <char> char\_arr(5, 'a', 'n', 'a', 'e', 'c');

    char char\_inp\_d[] = {'a', 'n', 'a', 'e', 'c'};

    ArrayContainer <char> char\_arr2(5, char\_inp\_d);

    char\_arr2.show\_data();

    char\_arr.check\_size();

    char\_arr.show\_data();

    ArrayContainer <int> int\_arr2(5, 1, 2, 3, 4, 5);

    char\_arr.at(3).show\_data();

    int\_arr.at(2).show\_data();

    int\_arr.check\_size();

    int\_arr.show\_data();

    int\_arr2.show\_data();

    ArrayContainer <bool> bool\_arr(3, true, false, true);

    bool\_arr.show\_data();

    int\_arr.resize(10);

    ArrayContainer <bool> arr(5, true);

    arr.show\_data();

    arr.get\_data\_type();

    ArrayContainer <bool> bool\_arr2(2, false, true);

    arr.copy(bool\_arr2);

    arr.show\_data();

    return 1;

}

// c-char i-int b-bool f-float d-double

**Вивід:**

A screenshot of a computer

Description automatically generated