Міністерство освіти і науки України

Чернівецький національний університет Імені Юрія Федьковича

Інститут фізико-технічних та комп’ютерних наук

Відділ комп’ютерних технологій

Кафедра математичних проблем управління і кібернетики

**ЗВІТ**

**про проходження**

**літньої обчислювальної практики, що**

**проходила з \_\_\_\_\_\_.2018 по \_\_\_\_\_\_\_\_\_**

**на базі кафедри МПУіК**

|  |  |
| --- | --- |
| Виконав студент | Бужак А.В. |
| Курс ІI |  |
| Група 241 |  |
| Керівник практики | Коцур М.П. |

Чернівці – 2018

**C#**

**Тема:** Файли.

**Мета роботи:** Розробка програм із використанням файлів.

*Завдання до лабораторної роботи:*

**Задача 1.** Робота із двійковими файлами.

4. Дана послідовність із n цілих чисел. Створити файл і записати в нього всі парні числа послідовності. Вивести вміст файлу на екран.

**Задача 2.** Робота з текстовим (символьним) файлом.

4. Даний текстовий файл. Знайти номер найдовшого рядка.

**Задача 3.** Для всіх. Текст у кутових дужка замінити відповідним чином.

Програмним шляхом:

1. У папці d:\temp створіть папки <прізвище\_студента>1 і <прізвище\_студента>2.

2. У папці <прізвище\_студента>1:

1. створіть файл t1.txt, у який запишіть наступний текст :

2. <Шевченко Андрій Іванович, 1990> року народження, місце проживання <м. Київ>

3. створіть файл t2.txt, у який запишіть наступний текст:

4. <Петренко Сергій Федорович, 1991 > року народження, місце проживання м. Чернівці

3. У папці <прізвище\_студента>2 створіть файл t3.txt, у який перепишіть спочатку текст із файлу t1.txt, а потім з t2.txt

4. Виведіть розгорнуту інформацію про створені файли.

5. Файл t2.txt перенесіть у папку < прізвище\_студента>2.

6. Файл t1.txt скопіюйте в папку < прізвище\_студента>2.

7. Папку < прізвище\_студента>2 перейменуйте в ALL, а папку < прізвище\_студента>1 вилучите.

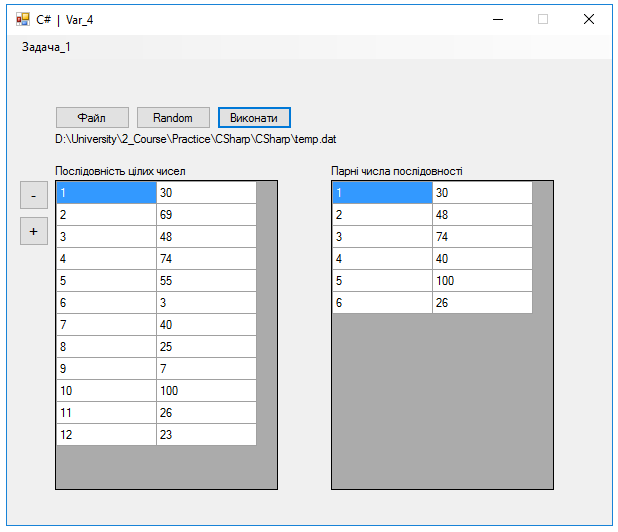
8. Вивести повну інформацію про файли папки All.

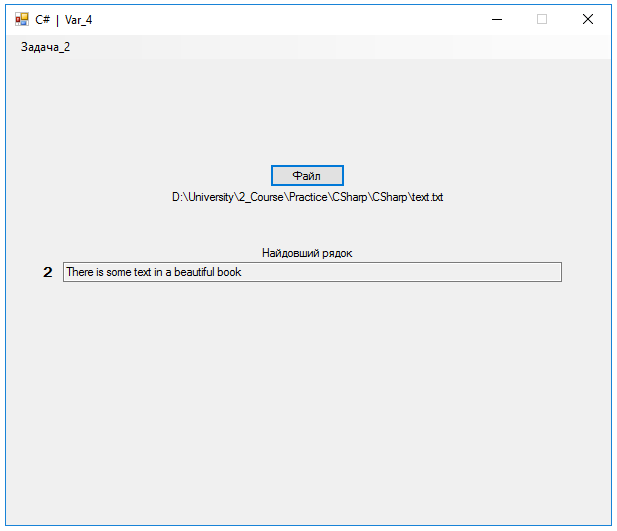
*Хід роботи:*

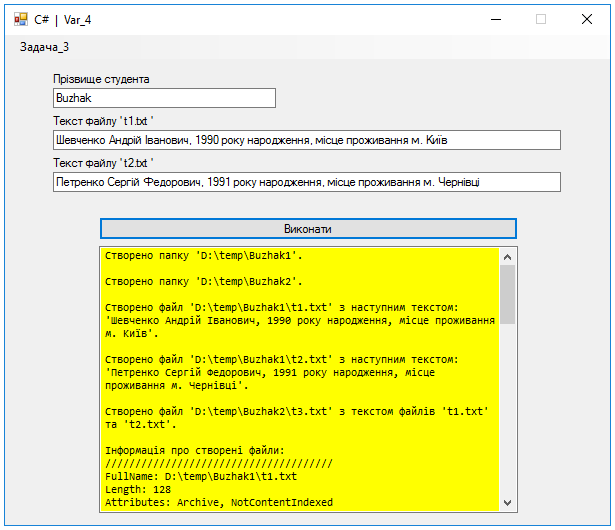
Для розв’язання поставлених задач створюю програму *C# Windows Forms*.

Вікно програми містить три контейнери *Panel* які можна перемикати в головному меню. Кожен з контейнерів містить набір елементів для роботи відповідної задачі.

Демонстрація роботи програми:







*Контрольні питання*

1) Що розуміється під терміном файл?

2) Тип файлів.

3) Класи роботи із файловою системою.

**Відповідь**

1) В інформатиці використовується наступне визначення: файл — це впорядкована сукупність даних, що зберігається на диску і займає іменовану область зовнішньої пам'яті.

Згідно з термінологією, прийнятою в програмуванні, файл — це довільний блок інформації, або пристрій вводу-виводу, асоційований із ним.

2) Розширення - це набір літер латинського алфавіту, цифр або спеціальних символів до 3 (або більше для ОС типу Windows), що визначає тип файлу. Оскільки розширення визначає тип файла, його ще називають просто типом. Як власне ім’я, так і розширення користувач може задавати довільно. Наприклад: *‘letter.txt’*, *‘ЛистПетрову.doc’*, *‘Собівартість.xls’*, *‘ДовідникЦін.dat’*.

3) У просторі імен *System.IO* передбачено чотири класи, які призначені для роботи з файловою системою комп'ютера: створення, видалення, перенесення і т.д. файлів і каталогів.

Перші два типи - *Directory* і *Fi1е* реалізують свої можливості за допомогою статичних методів, тому дані класи можна використовувати без створення відповідних об'єктів (екземплярів класів).

Наступні типи - *DirectoryInfo* і *FileInfo* мають схожі функціональні можливості c *Directory* і *Fi1е*, але породжені від класу *FileSystemInfo* і тому реалізуються шляхом створення відповідних екземплярів класів.

**C++/CLR**

**Тема:** Створення *Windows Form.*

**Мета роботи:** Ознайомлення з функціями класів: деструктори, індексатори, операції класу, операції перетворення типів.

*Завдання до лабораторної роботи:*

Написати програму з використання *Windows Form*  виведення малюнка, згідно варіанту. В програмі розробити форму для ведення даних про об’єкти малюнка (тип об’єкта, розміри, колір тощо) та для виведення малюнка в об’єкт *Image*. Малюнок будується як набір елементів(типи елементів задаються згідно варіанту), координати розташування елемента в об’єкт *PictureBox* задавати за допомогою датчика випадкових чисел в межах полотна об’єкт *PictureBox*. Кожний елемент є об’єктом одного з похідних класів. Набір – задається, як масив на абстрактний базовий клас. В базовому класі передбачити віртуальні функцію малювання, функцію переміщення та інші функції. Базовий клас фігура похідні класи :

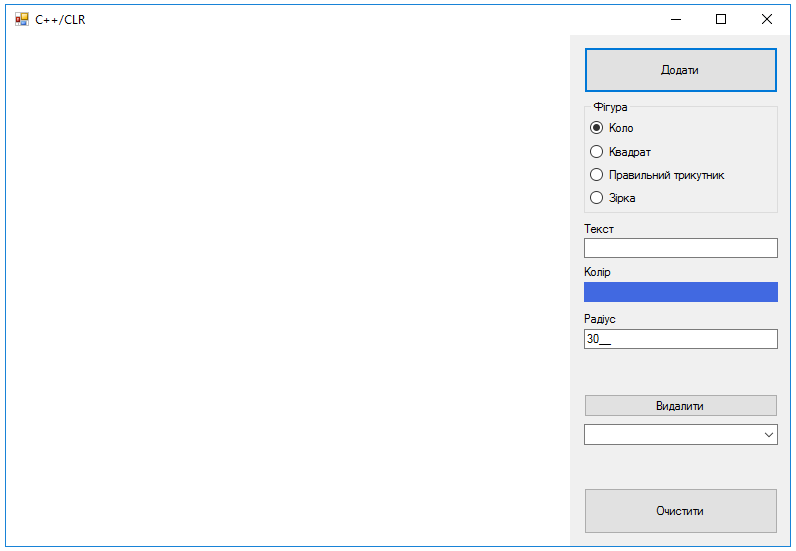
1.4. Коло, квадрат, правильний трикутник та зірку.

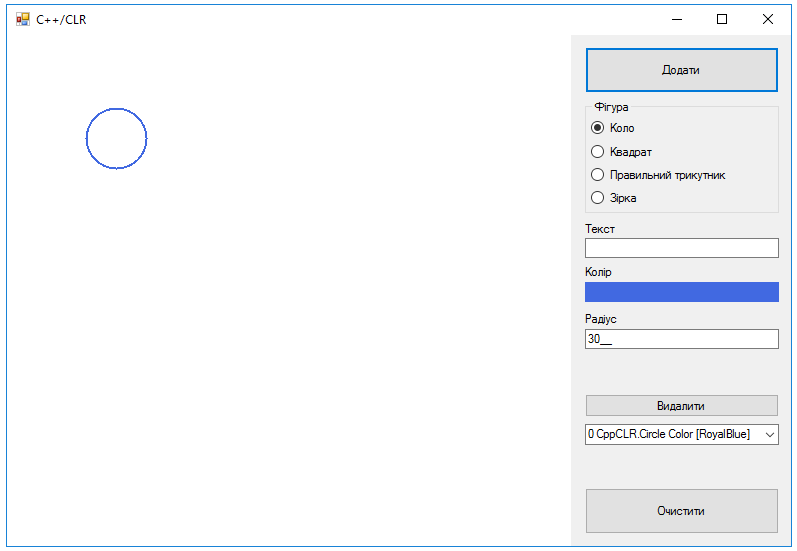
*Хід роботи:*

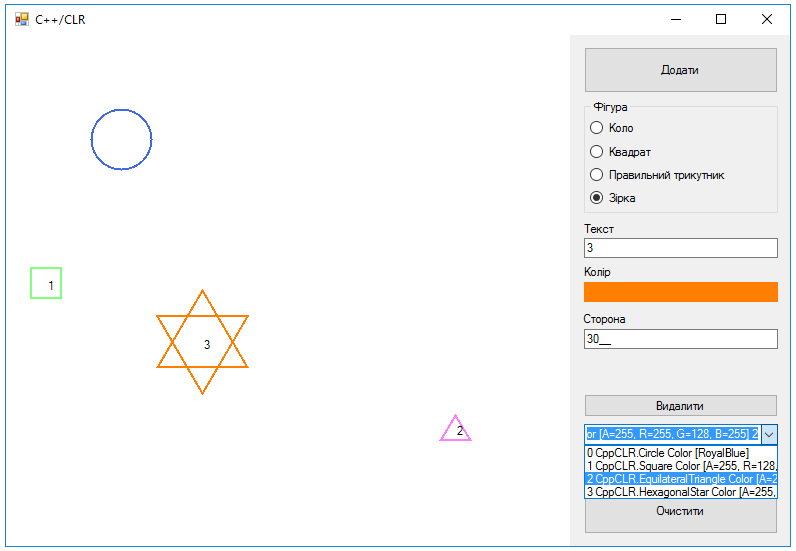
Для розв’язання поставленої задачі створюю програму *C++/CLR Windows Forms*.

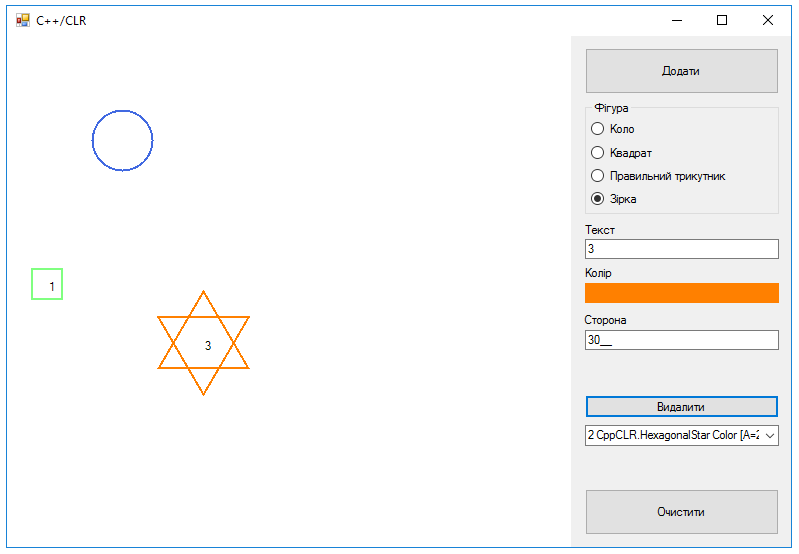
Вікно програми містить контейнер *Panel* з елементами керування та елемент *PictureBox* для відображення фігур. *Panel* прикріплений до правого краю вікна, а *PictureBox* заповнює всю вільну область вікна залежно від його поточного розміру.

Демонстрація роботи програми:









**C++**

**Тема:** Обробка виключних ситуацій мови С++.

*Завдання до лабораторної роботи:*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Таблиця 4** |  |  | **Таблиця 5** |  |  | **Таблиця 6** |
| **x** | **U(x)** |  | **x** | **T(x)** |  | **text** | **x** |
| -5,0000 | 0,2801 |  | -10,0000 | 0,7832 |  | aet | 1,175 |
| -4,5000 | 0,2093 |  | -9,0000 | 1,1063 |  | bet | 1,278 |
| -4,0000 | 0,6190 |  | -8,0000 | 1,2486 |  | cet | 1,381 |
| -3,5000 | 0,8811 |  | -7,0000 | 1,1587 |  | set | 1,484 |
| -3,0000 | 1,0422 |  | -6,0000 | 0,9105 |  | get | 1,587 |
| -2,5000 | 1,1463 |  | -5,0000 | 0,2801 |  | ret | 1,69 |
| -2,0000 | 1,2176 |  | -4,0000 | 0,6190 |  | het | 1,793 |
| -1,5000 | 1,2560 |  | -3,0000 | 1,0422 |  | met | 1,896 |
| -1,0000 | 1,1998 |  | -2,0000 | 1,2176 |  | net | 1,999 |
| -0,5000 | 1,1209 |  | -1,0000 | 1,1998 |  | qet | 2,102 |
| 0,0000 | 1,0039 |  | 0,0000 | 1,0039 |  | tet | 2,205 |
| 0,5000 | 0,8196 |  | 1,0000 | 0,5187 |  | wet | 2,308 |
| 1,0000 | 0,5187 |  | 2,0000 | 0,4054 |  | yet | 2,411 |
| 1,5000 | 0,0707 |  | 3,0000 | 0,9603 |  | iet | 2,514 |
| 2,0000 | 0,4054 |  | 4,0000 | 1,1803 |  | oet | 2,617 |
| 2,5000 | 0,7487 |  | 5,0000 | 1,2338 |  | pet | 2,72 |
| 3,0000 | 0,9603 |  | 6,0000 | 1,0761 |  | det | 2,823 |
| 3,5000 | 1,0926 |  | 7,0000 | 0,7068 |  | fet | 2,926 |
| 4,0000 | 1,1803 |  | 8,0000 | 0,1450 |  | let | 3,029 |
| 4,5000 | 1,2418 |  | 9,0000 | 0,8533 |  | zet | 3,132 |
| 5,0000 | 1,2338 |  | 10,0000 | 1,1347 |  | vet | 3,235 |

**Задача 4.** Задано текстові файли ***dat1.dat, dat2.dat*** та ***dat3.dat*** , які містить інформацію таблиця 4, таблиця 5 та таблиця 6 відповідно, та дійсні змінні ***x, y*** та ***z***, текстовий рядок ***text*** які вводяться із стандартного потоку введення. Обчислити значення функції ***Variant(r,k)***,яказнаходитьсяза алгоритмами у порядку пріоритету.

**Алгоритм 1.**

1. 
2. 
3. 
4. 
5. 
6. Функції та  за даними, які вводяться, з файлів відповідно ***dat1.dat, dat2.dat***.
   1. Якщо файл ***dat1.dat*** не відкривається, або відсутній, тоді функцію  порахувати за Алгоритмом 2;
   2. Якщо , тоді функцію  порахувати за Алгоритмом 2;
   3. Якщо , шукаємо  та , такі що , тоді   
      ;
   4. Якщо файл ***dat2.dat*** не відкривається, або відсутній функцію  порахувати за Алгоритмом 3.
   5. Якщо , тоді функцію  порахувати за Алгоритмом 2;
   6. Якщо , шукаємо  та , такі що , тоді   
      .
7. 
8. Функція  обчислює максимальне значення серед заданих параметрів.
9. 

Умова  , означає що текстовий рядок порожній.

1. Для обчислення функції  відкриваємо файл ***dat3.dat.*** 
   1. Якщо файл ***dat3.dat*** не відкривається, або відсутній, тоді видати відповідне повідомлення, що неможна відкрити файл та вийти з програми;
   2. Здійснити пошук відповідного слова.
   3. Якщо слово знайдено повернути значення яке відповідає заданому слову у файлі;
   4. Якщо слово відсутнє повернути значення нуль.
2. ;
3. 
   1. Якщо , тоді змінні  присвоїти 0;
4. 
5. 
   1. Якщо , тоді змінні  присвоїти 0;
   2. Якщо , тоді змінні  присвоїти 0;

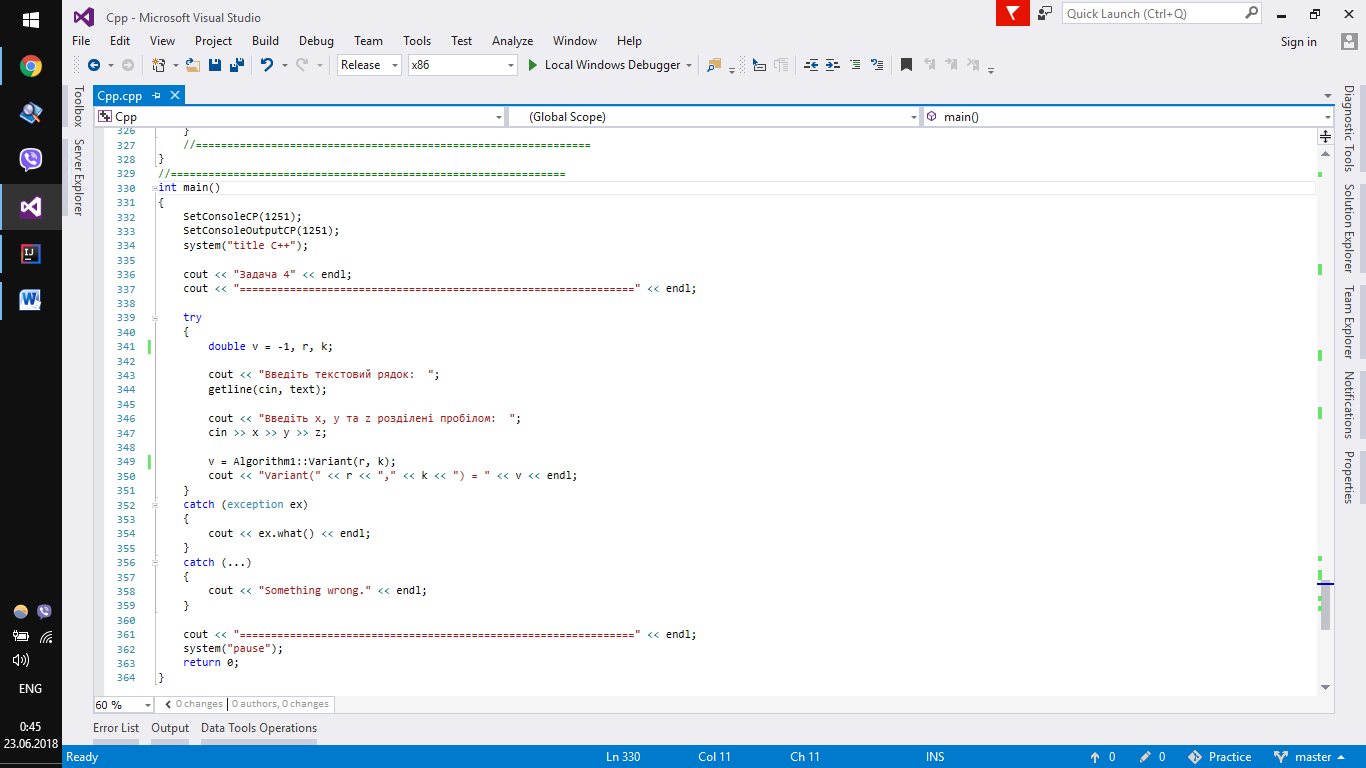
**Алгоритм 2.**

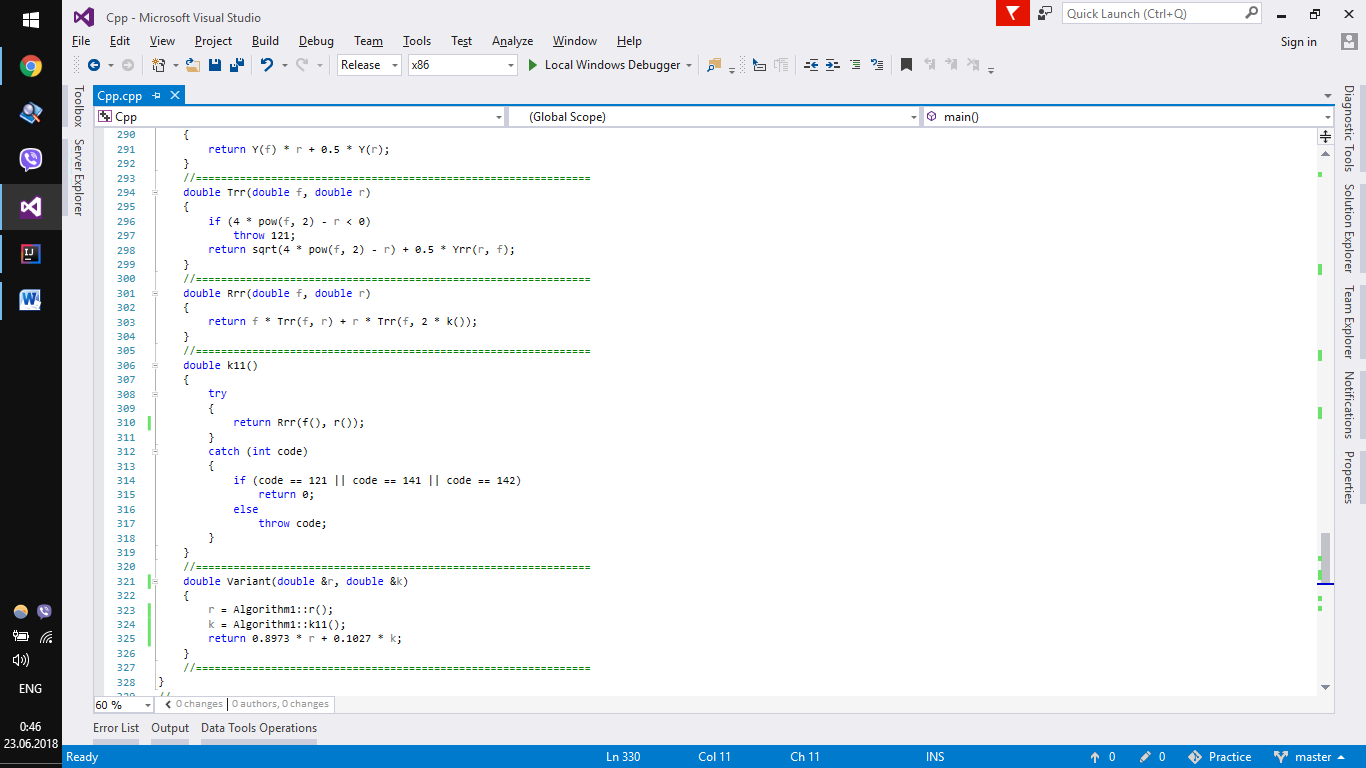
1. 
2. 
3. 
4. 
5. 

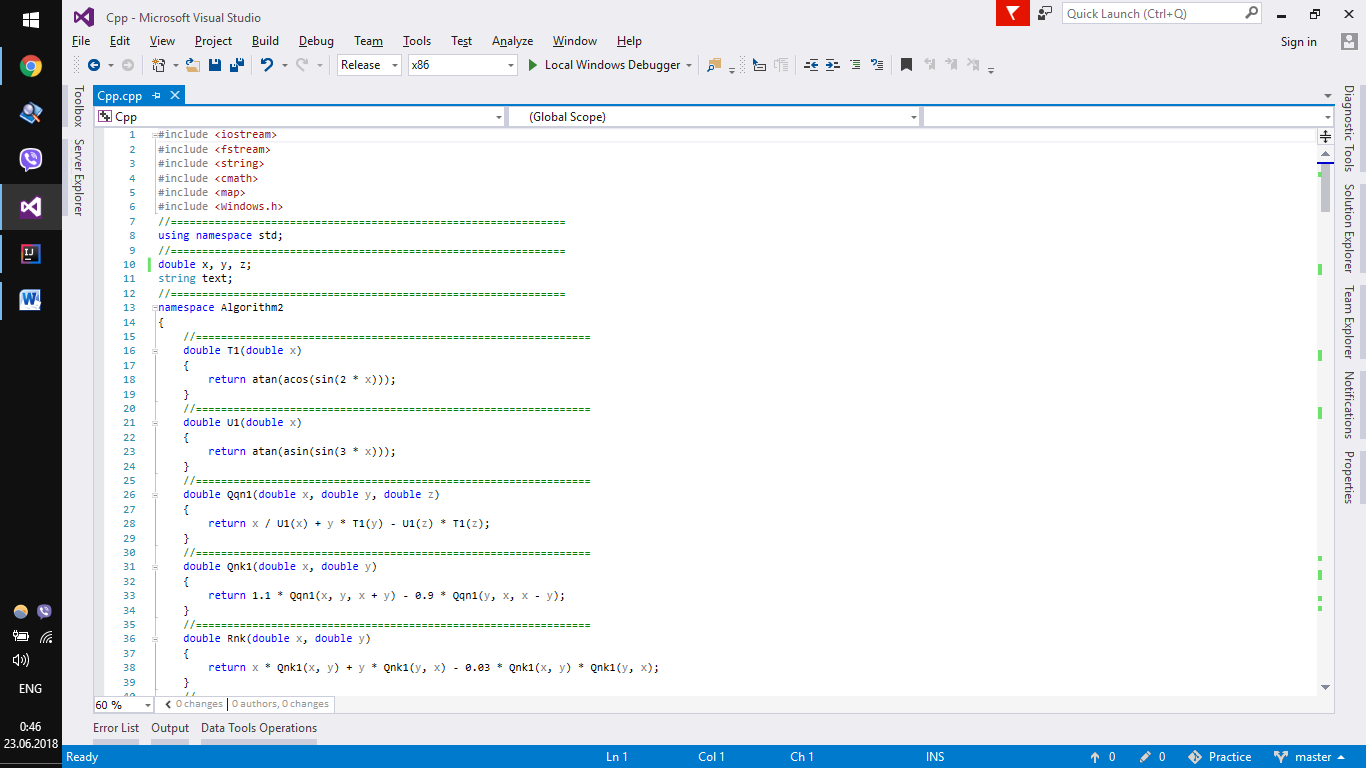
**Алгоритм 3.**

1. 
2. 
3. 
4. 
5. 

*Фрагменти коду розробленої програми:*







*Демонстрація роботи програми:*





**Java**

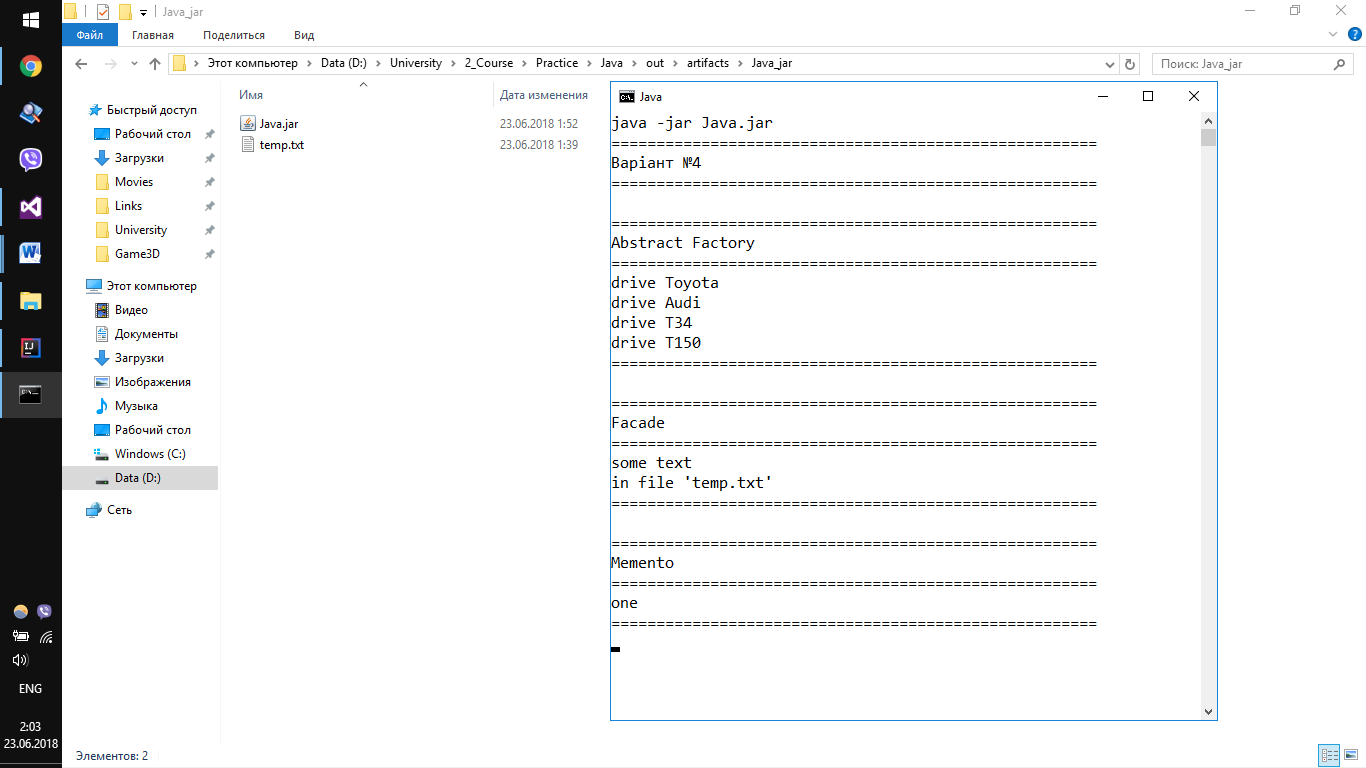
**Тема:** Шаблони проектування.

**Мета роботи:** Вивчення шаблонів мови програмування *Java* .

*Завдання до лабораторної роботи:*

|  |
| --- |
| Варіант № 4. |
| 1. Abstract Factory (абстрактна фабрика). |
| 2. Facade (фасад). |
| 3. Memento (зберігач). |

*Демонстрація виконання програми:*



*Модулі коду розробленої програми:*

**Файл ‘Main.java’**

**import** java.io.IOException;  
**public class** Main {  
 **public static void** main(String[] args) {  
  
 System.***out***.println(**"======================================================"**);  
 System.***out***.println(**"Варiант №4"**);  
 System.***out***.println(**"======================================================"**);  
 System.***out***.println();  
 System.***out***.println(**"======================================================"**);  
 System.***out***.println(**"Abstract Factory"**);  
 System.***out***.println(**"======================================================"**);  
  
  
 Factory carFactory = **new** AbstractFactory().createFactory(**"Car"**);  
 Factory tankFactory = **new** AbstractFactory().createFactory(**"Tank"**);  
  
 Car toyota = carFactory.createCar(**"Toyota"**);  
 Car audi = carFactory.createCar(**"Audi"**);  
  
 toyota.drive();  
 audi.drive();  
  
 Tank t34 = tankFactory.createTank(**"T34"**);  
 Tank t150 = tankFactory.createTank(**"T150"**);  
  
 t34.drive();  
 t150.drive();  
  
 System.***out***.println(**"======================================================"**);  
 System.***out***.println();  
  
 System.***out***.println(**"======================================================"**);  
 System.***out***.println(**"Facade"**);  
 System.***out***.println(**"======================================================"**);  
 FileReadFacade fileReadFacade = **new** FileReadFacade();  
 **try** {  
 System.***out***.println(fileReadFacade.readFile(**"temp.txt"**));  
 }  
 **catch** (IOException ex) {  
 System.***out***.println(ex.getMessage());  
 }  
 System.***out***.println(**"======================================================"**);  
 System.***out***.println();  
  
 System.***out***.println(**"======================================================"**);  
 System.***out***.println(**"Memento"**);  
 System.***out***.println(**"======================================================"**);  
 Originator originator = **new** Originator();  
 originator.setState(**"one"**);  
 CareTaker careTaker = **new** CareTaker();  
 careTaker.setMemento(originator.createMemento());  
 originator.setState(**"two"**);  
 originator.getDataFromMemento(careTaker.getMemento());  
 System.***out***.println(originator.getState());  
 System.***out***.println(**"======================================================"**);  
 }  
}

**Файл ‘AbstractFactoryPattern.java’**

**interface** Car {  
 **void** drive();  
}  
  
**class** Toyota **implements** Car {  
 **public void** drive() {  
 System.***out***.println(**"drive Toyota"**);  
 }  
}  
  
**class** Audi **implements** Car {  
 **public void** drive() {  
 System.***out***.println(**"drive Audi"**);  
 }  
}  
  
**class** CarFactory **implements** Factory {  
 **public** Car createCar(String typeOfCar) {  
 **switch** (typeOfCar) {  
 **case "Toyota"** : **return new** Toyota();  
 **case "Audi"** : **return new** Audi();  
 **default** : **return null**;  
 }  
 }  
  
 **public** Tank createTank(String typeOfTank) {  
 **return null**;  
 }  
}  
  
**interface** Tank {  
 **void** drive();  
}  
  
**class** T34 **implements** Tank {  
 @Override  
 **public void** drive() {  
 System.***out***.println(**"drive T34"**);  
 }  
}  
  
**class** T150 **implements** Tank {  
 @Override  
 **public void** drive() {  
 System.***out***.println(**"drive T150"**);  
 }  
}  
  
**class** TankFactory **implements** Factory {  
 **public** Tank createTank(String typeOfTank) {  
 **switch** (typeOfTank) {  
 **case "T34"** : **return new** T34();  
 **case "T150"** : **return new** T150();  
 **default** : **return null**;  
 }  
 }  
  
 **public** Car createCar(String typeOfCar) {  
 **return null**;  
 }  
}  
  
**interface** Factory {  
 Car createCar(String typeOfCar);  
 Tank createTank(String typeOfTank);  
}  
  
**class** AbstractFactory {  
 Factory createFactory(String typeOfFactory) {  
 **switch** (typeOfFactory) {  
 **case "Car"** : **return new** CarFactory();  
 **case "Tank"** : **return new** TankFactory();  
 **default** : **return null**;  
 }  
 }  
}

**Файл ‘FacadePattern.java’**

**import** java.io.\*;  
**class** FileReadFacade {  
 String readFile(String fileName) **throws** IOException {  
 BufferedReader reader = **new** BufferedReader(**new** FileReader(fileName));  
 StringBuilder stringBuilder = **new** StringBuilder();  
 **int** j = 0;  
 **while** ((j = reader.read()) != -1) {  
 stringBuilder.append((**char**)j);  
 }  
 **return** stringBuilder.toString();  
 }  
}

**Файл ‘MementoPattern.java’**

**class** CareTaker {  
 Memento **memento**;  
 **public** Memento getMemento() {  
 **return memento**;  
 }  
 **public void** setMemento(Memento memento) {  
 **this**.**memento** = memento;  
 }  
}  
**class** Originator {  
 String **state**;  
 **public** String getState() { **return state**; }  
 **public void** setState(String state) { **this**.**state** = state; }  
 Memento createMemento() {  
 **return new** Memento(**state**);  
 }  
 **void** getDataFromMemento(Memento memento) {  
 **this**.**state** = memento.getState();  
 }  
}  
**class** Memento {  
 String **state**;  
 **public** Memento(String state) {  
 **this**.**state** = state;  
 }  
 **public** String getState() { **return state**; }  
}