МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

Національний технічний університет України

"Київський політехнічний інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки

Лабораторна робота №3

з дисципліни «Об'єктно оріентовне програмування»

Тема: «Шаблони проектування. Поведінкові шаблони»

Виконав:

студенти групи ІП-81

Балачін Петро

Київ-2020

**Мета роботи**: ознайомитися з основними шаблонами проектування, навчитися застосовувати їх при проектуванні і розробці ПЗ.

**Хід роботи:**

**Варіант 3**

class Program

{

static void Main(string[] args)

{

bool RUN = true;

while (RUN)

{

Context context = new Context();

Console.WriteLine("Write elements of A plurality: ");

string inputA = Console.ReadLine();

Console.WriteLine("Write elements of B plurality: ");

string inputB = Console.ReadLine();

Console.WriteLine("Write elements of U plurality: ");

string inputU = Console.ReadLine();

List<string> A = inputA.Split(" ").ToList();

List<string> B = inputB.Split(" ").ToList();

List<string> U = inputU.Split(" ").ToList();

Console.WriteLine("Operation : ");

string inputOP = Console.ReadLine();

if (inputOP == "AND")

{

context.setOperation(new AND\_Operation());

}

else if (inputOP == "OR")

{

context.setOperation(new OR\_Operation());

}

else if (inputOP == "EQUALS")

{

context.setOperation(new EQUALS\_Operation());

}

else if (inputOP == "NOT")

{

context.setOperation(new NOT\_Operation());

}

else if (inputOP == "CONTAINS")

{

context.setOperation(new CONTAINS\_Operation());

}

else if (inputOP == "COMPARISON")

{

context.setOperation(new COMPARISON\_Operation());

}

else if (inputOP == "COMPOUND")

{

context.setOperation(new COMPOUND\_Operation());

}

else

{

Console.WriteLine("NO SUCH A OPERATION");

}

List<string> result = context.executeOperation(A, B, U);

Console.WriteLine("RESULT: ");

Console.Write("( ");

foreach (string el in result)

{

Console.Write(el + " ");

}

Console.WriteLine(" )");

Console.Clear();

}

}

}

abstract class Operation

{

public abstract List<string> execute(List<string> A, List<string> B, List<string> U);

}

class AND\_Operation : Operation

{

public override List<string> execute(List<string> A, List<string> B, List<string> U)

{

List<string> C = new List<string>();

foreach(string a in A)

{

foreach(string b in B)

{

if(a == b && !C.Contains(a))

{

C.Add(a);

}

}

}

return C;

}

}

class OR\_Operation : Operation

{

public override List<string> execute(List<string> A, List<string> B, List<string> U)

{

List<string> C = new List<string>();

foreach (string a in A)

{

C.AddRange(A);

C.AddRange(B);

C = C.Distinct().ToList();

}

return C;

}

}

class EQUALS\_Operation : Operation

{

public override List<string> execute(List<string> A, List<string> B, List<string> U)

{

bool A\_EQUALS(string s)

{

return B.Contains(s);

}

bool B\_EQUALS(string s)

{

return A.Contains(s);

}

List<string> C = new List<string>();

if (A.TrueForAll(A\_EQUALS) && B.TrueForAll(B\_EQUALS))

{

C.Add("True");

}

else

{

C.Add("False");

}

return C;

}

}

class NOT\_Operation : Operation

{

public override List<string> execute(List<string> A, List<string> B, List<string> U)

{

List<string> C = new List<string>();

C = U.Except(A).ToList();

return C;

}

}

class CONTAINS\_Operation : Operation

{

public override List<string> execute(List<string> A, List<string> B, List<string> U)

{

bool Contains(String s)

{

return B.Contains(s);

}

List<string> C = new List<string>();

if (A.TrueForAll(Contains))

{

C.Add("True");

}

else

{

C.Add("False");

}

return C;

}

}

class COMPARISON\_Operation : Operation

{

public override List<string> execute(List<string> A, List<string> B, List<string> U)

{

List<string> C = new List<string>();

A = A.Distinct().ToList();

B = B.Distinct().ToList();

if(A.Count() > B.Count())

{

C.Add("A > B");

}

else if (A.Count() < B.Count())

{

C.Add("A < B");

}

else

{

C.Add("A and B are even");

}

return C;

}

}

class COMPOUND\_Operation : Operation

{

public override List<string> execute(List<string> A, List<string> B, List<string> U)

{

List<string> C = new List<string>();

C.AddRange(A);

C.AddRange(B);

return C;

}

}

class Context

{

private Operation operation;

public void setOperation(Operation op)

{

operation = op;

}

public List<string> executeOperation(List<string> A, List<string> B, List<string> U)

{

return operation.execute(A,B,U);

}

}

**Висновок:**Я засвоїв теоретичний матеріал з відповідної теми та закріпив його,виконавши практичні завдання.