МІНІСТЕРСТВО  ОСВІТИ  І  НАУКИ  УКРАЇНИ

НАЦІОНАЛЬНИЙ   ТЕХНІЧНИЙ   УНІВЕРСИТЕТ   УКРАЇНИ

“КИЇВСЬКИЙ  ПОЛІТЕХНІЧНИЙ  ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**Лабораторна робота №** 2

з дисципліни “Програмування”

тема “ C# .Net. Розширені можливості реалізації ООП у мові C#. Події.”

|  |  |  |
| --- | --- | --- |
| Виконав(ла)  студент(ка) II курсу  групи КП-01  Грабовська Анастасія Дмитрівна  (*прізвище, ім’я, по батькові*) |  | Перевірив  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладач  Заболотня Тетяна Миколаївна  (*прізвище, ім’я, по батькові*) |

Київ 2021

**Мета роботи**

Ознайомитися з такими можливостями мови програмування C# як абстрактні класи, інтерфейси, делегати. Вивчити механізми оброблення подій у C#, а також можливості, які мають методи-розширення.

**Постановка задачі**

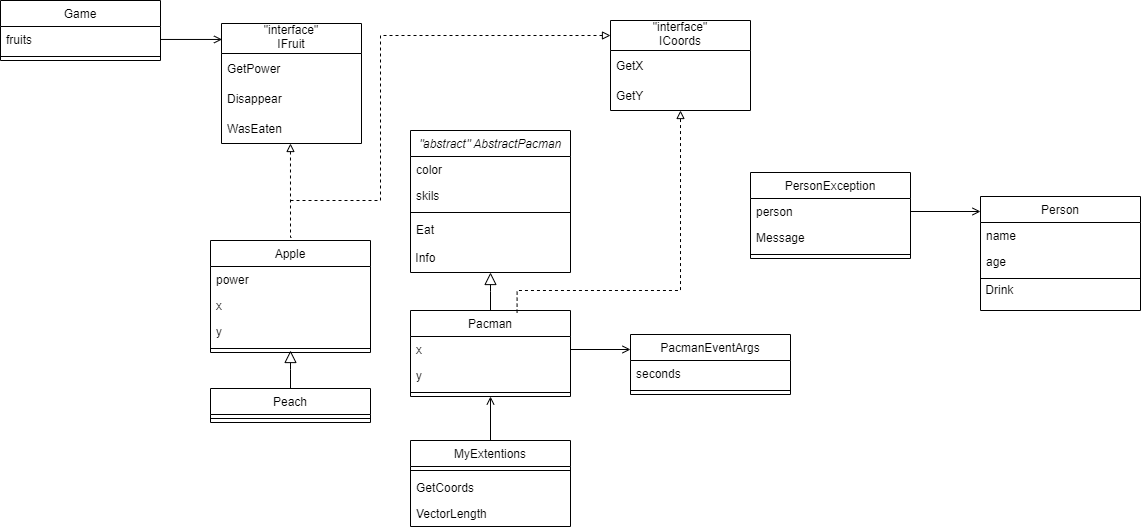
Для ієрархії класів, побудованої в лабораторній роботі №1, реалізувати:

1. Множину інтерфейсів. При чому один з класів повинен реалізовувати щонайменше 2 інтерфейси. Також продемонструвати реалізацію explicit implementation інтерфейса, обгрунтувати її використання **(1 бал)**.
2. Абстрактний клас. Забезпечити його наслідування. Наявність в цьому класі абстрактних методів - обов’язкова **(1 бал)**.
3. Механізм «делегат – подія – обробник події» **(2 бали)**.
4. Перетворити код, який забезпечує роботу з подіями та обробниками подій, на код, що використовує (\*) **(2 бали)**:
   1. анонімні методи;
   2. lambda-вирази;
   3. типи Action та Func (кожен з них).

(\*) - допускається реалізація коду однієї події різними способами, необов’язково різних подій.

1. Механізм створення та оброблення власних помилок **(2 бали)**:
   1. створити новий клас виключної ситуації;
   2. створити новий клас аргументів для передачі їх до обробника виключної ситуації;
   3. забезпечити ініціювання створеної виключної ситуації та продемонструвати, як працює обробник даної помилки;
   4. реалізувати різні сценарії оброблення помилки.
2. Метод-розширення будь-якого класу **(1 бал)**.

**UML діаграма класів**



**Фрагменти коду:**

1.

|  |
| --- |
| interface IFruit      {          int GetPower();          void Disappear();          void WasEaten(Pacman pacman, PacmanEventArgs args);      }      interface ICoords      {          int GetX();          int GetY();      } |
| public virtual void Disappear()          {              WriteLine($"Apple with power {power} was disappeared in coordinates ({x}, {y}).");          }          void IFruit.Disappear()          {              WriteLine("Some fruit was disappeared.");          } |

2.

|  |
| --- |
| abstract class AbstrackPacman      {          public string color;          protected int skils;          public abstract void Eat(IFruit fruit);          public abstract void Info();      } |

3.

|  |
| --- |
| delegate void PacmanHandle(Pacman pacman, PacmanEventArgs args);      class PacmanEventArgs : EventArgs      {          public int seconds;          public PacmanEventArgs(int seconds)          {              this.seconds = seconds;          }      }      class Pacman : AbstrackPacman, ICoords      {          public event PacmanHandle PacmanEatEvent;          ...          public override void Eat(IFruit fruit)          {              skils+=fruit.GetPower();              PacmanEventArgs args = new PacmanEventArgs(13);              PacmanEatEvent((Pacman)this, args);          }      }      class Apple : IFruit, ICoords      {          ...          public virtual void WasEaten(Pacman pacman, PacmanEventArgs args)          {              if (args.seconds < 10)                  Console.WriteLine("Apple was eaten very quickly by " + pacman.color + " pacman.");              else                  Console.WriteLine("Apple wasn't eaten very quickly by " + pacman.color + " pacman.");          }      }      class Peach : Apple      {  ...          public override void WasEaten(Pacman pacman, PacmanEventArgs args)          {              if (args.seconds < 20)                  Console.WriteLine("Peach was eaten very quickly by " + pacman.color + " pacman.");              else                  Console.WriteLine("Peach wasn't eaten very quickly by " + pacman.color + " pacman.");          }      }      class Game      {          IFruit[] fruits;          public Game(Pacman pacman)          {              fruits = new IFruit[2];              fruits[0] = new Apple(3, 7);              fruits[1] = new Peach(1, 2);              foreach (IFruit f in fruits)                  pacman.PacmanEatEvent += new PacmanHandle(f.WasEaten);          }      } |
| Pacman pacman = new Pacman("red", 7, 5);              Game game = new Game(pacman);              pacman.Eat(a); |

4.

|  |
| --- |
| PacmanEventArgs pacmanArgs = new PacmanEventArgs(30);              PacmanHandle pacmanHandle = delegate(Pacman pacman, PacmanEventArgs pacmanArgs)              {                  if (pacmanArgs.seconds < 50)                      Console.WriteLine("Anonymous method by " + pacman.color + " pacman was very quickly.");                  else                      Console.WriteLine("Anonymous method by " + pacman.color + " pacman wasn't very quickly.");              };              pacmanHandle(pacman, pacmanArgs);              Action<string> line = name => Console.WriteLine("{0} invoked an action", name);              Func<int, int, string> result = (a, b) => $"{a} + {b} is {a+b}";              line("Anastasia");              WriteLine(result(3,8)); |

5.

|  |
| --- |
| class Person      {          public string name;          public int age;          public Person(string name, int age)          {              this.name = name;              this.age = age;          }          public void Drink()          {              if(age < 18) throw new PersonException(this);              WriteLine($"{this.name} drink vodka.");          }      }      class PersonException : Exception      {          public Person args;          public PersonException(Person args) : base()          {              this.args = args;          }          public override string Message => $"Person {args.name} is too young. Come back in {18 - args.age}";      } |
| Person p1 = new Person("John", 16);              try              {                  p1.Drink();              }              catch(PersonException e)              {                  WriteLine(e.Message);              }              finally              {                  Person p2 = new Person("Mike", 19);                  p2.Drink();              }                  // Person John is too young. Come back in 2                  // Mike drink vodka. |

6.

|  |
| --- |
| static class MyExtentions      {          public static double VectorLength(this Pacman pacman)          {              return Math.Sqrt(pacman.GetX()\*pacman.GetX()+pacman.GetY()\*pacman.GetY());          }          public static void GetCoords(this Pacman pacman)          {              WriteLine($"This is an extention method. Pacman has coordinates ({pacman.GetX()}, {pacman.GetY()}).");          }      } |

**Висновки:**

Під час виконання лабораторної роботи я ознайомилася з такими можливостями мови програмування C# як абстрактні класи, інтерфейси, делегати. Також я вивчила механізми оброблення подій у C# та мож можливості, які мають методи-розширення.