**Лабораторная работа №14 Часть 2**

*«Обработка исключительных ситуаций»*

**Рекомендации по программированию**

* Поля в классе необходимо делать private.
* Интерфейс должен быть полным, т.е. предоставлять возможность выполнять любые разумные действия с классом. И одновременно минимально необходимым, т.е. без дублирования и пересечения возможностей методов.
* Каждый метод класса должен решать только одну задачу. Не стоит объединять два коротких независимых фрагмента кода в один метод.
* Если действия встречаются хотя бы дважды, стоит оформить их в отдельные функции.
* Все заданные методы, поля и экземпляры классов должны иметь осмысленные имена.
* Каждое имя интерфейса должно начинаться с буквы I.

**Требования к лабораторной**

* Каждый разрабатываемый класс должен, как правило, содержать следующие элементы: поля с различными спецификаторами, конструкторы с параметрами и без параметров, методы, свойства.
* Методы и свойства должны обеспечивать непротиворечивый, полный, минимальный и удобный интерфейс класса.
* При возникновении ошибок должны выбрасываться исключения

**Постановка задачи**

**Задача 2:** Задан некоторый класс с полями согласно индивидуальному заданию. Реализовать для данного класса конструктор с параметрами и без. Задать и реализовать методы для этого класса, определенные в индивидуальном задании. Позволить пользователю вводить данные с клавиатуры в любые поля. Задать не менее пяти объектов класса. С помощью механизмов обработки исключений реализовать методы класса в указанных в индивидуальном задании ограничениях. В случае ввода не верных значений вывести соответствующее значение, номер строки в коде, название проекта и название функции, в которой допущена ошибка.

|  |  |  |  |
| --- | --- | --- | --- |
| **Вариант** | **Класс** | **Поля и ограничения** | **Методы** |
| 1. | Студент колледжа | |  |  | | --- | --- | | Имя | С большой буквы | | Фамилия | С большой буквы | | Возраст | От 16 до 21 года | | Итоговые оценки за семестр | От 0 до 10 целые | | Количество прогулов | От 0 до 333 целые | | * Ввода-вывода данных * Определения размера стипендии  |  |  |  | | --- | --- | --- | | Кол-во прогулов | Средняя | Сумма | | <10 | >=5 | 50 р | | <7 | >=7 | 75 р | | <3 | >=9 | 100 р | |

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab14

{

class Student

{

string name;

string surName;

int age;

int mark;

int proguly;

public Student(string name, string surName, int age, int mark, int proguly)

{

this.name = name;

this.surName = surName;

this.age = age;

this.mark = mark;

this.proguly = proguly;

}

public string Name { get => name; set => name = value; }

public string SurName { get => surName; set => surName = value; }

public int Age { get => age; set => age = value; }

public int Mark { get => mark; set => mark = value; }

public int Proguly { get => proguly; set => proguly = value; }

public static Student Add()

{

Console.WriteLine("Введите имя");

string name = Console.ReadLine();

if (name[0].ToString() != name[0].ToString().ToUpper())

{

throw new NameException();

}

Console.WriteLine("Введите фамилию");

string surName = Console.ReadLine();

if (surName[0].ToString() != surName[0].ToString().ToUpper())

{

throw new SurNameException();

}

Console.WriteLine("Введите возраст");

int age = Convert.ToInt32(Console.ReadLine());

if (age < 16 || age > 21)

{

throw new AgeException();

}

Console.WriteLine("Введите оценку");

int mark = Convert.ToInt32(Console.ReadLine());

if (mark < 0 || mark > 10)

{

throw new MarkException();

}

Console.WriteLine("Введите прогулы");

int proguly = Convert.ToInt32(Console.ReadLine());

if (proguly < 0 || proguly > 333)

{

throw new ProgulyException();

}

return new Student(name, surName, age, mark, proguly);

}

public void ShowInfo()

{

Console.WriteLine(Name + " " + SurName + " " + Age + " " + Mark + " " + Proguly);

}

public double GetStepuha()

{

if (proguly < 10 && mark>=5)

{

return 50;

}

if (proguly < 7 && mark >= 7)

{

return 75;

}

if (proguly < 3 && mark >= 9)

{

return 100;

}

return 0;

}

}

class NameException : Exception

{

public NameException()

{

Console.WriteLine("Имя с маленькой");

}

}

class SurNameException : Exception

{

public SurNameException()

{

Console.WriteLine("Фамилия с маленькой");

}

}

class AgeException : Exception

{

public AgeException()

{

Console.WriteLine("Возраст введен не верно");

}

}

class MarkException : Exception

{

public MarkException()

{

Console.WriteLine("Оценка введена не верно");

}

}

class ProgulyException : Exception

{

public ProgulyException()

{

Console.WriteLine("Прогулы введены не верно");

}

}

}

