# Exercise: Intro and Basic Syntax

<https://softuni.bg/trainings/live/details?trainingLabId=95>

Video Exersizes !

<https://softuni.bg/trainings/resources/video/57199/video-15-january-2021-atanas-atanasov-csharp-fundamentals-january-2021/3213>

Problems for exercises and homework for the ["C# Fundamentals" course @ SoftUni](https://softuni.bg/trainings/2363/csharp-fundamentals-may-2019)  
You can check your solutions in [Judge](https://judge.softuni.bg/Contests/1204)

## Ages

Write a program that determines whether based on the given age a person is: baby, child, teenager, adult, elder. The bounders are:

* **0-2 – baby;**
* **3-13 – child;**
* **14-19 – teenager;**
* **20-65 – adult;**
* **>=66 – elder;**
* All the values are **inclusive**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 20 | adult |
| 1 | baby |
| 100 | elder |

## Divison

You will be given an integer and you have to print on the console whether that number is divisible by the following numbers: 2, 3, 6, 7, 10. You should **always take the bigger division**. If the number is divisible by both **2** and **3** it is also divisible by **6** and you should print only the division by 6. If a number is divisible by **2** it is sometimes also divisible by **10** and you should print the division by 10. If the number is not divisible by any of the given numbers print “**Not divisible”.** Otherwise print “**The number is divisible by {number}**”.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 30 | The number is divisible by 10 |
| 15 | The number is divisible by 3 |
| 12 | The number is divisible by 6 |
| 1643 | Not divisible |

## Vacation

You are given a group of people, type of the group, on which day of the week they are going to stay. Based on that information calculate how much they have to pay and print that price on the console. Use the table below. In each cell is the price for a **single person**. The output should look like that: “**Total price: {price}**”. The price should be formatted to the second decimal point.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Friday** | **Saturday** | **Sunday** |
| **Students** | 8.45 | 9.80 | 10.46 |
| **Business** | 10.90 | 15.60 | 16 |
| **Regular** | 15 | 20 | 22.50 |

There are also discounts based on some conditions:

* **Students –** if the group is bigger than or equal to 30 people you should reduce the **total** price by 15%
* **Business –** if the group is bigger than or equal to 100 people **10** of them can stay **for free.**
* **Regular –** if the group is bigger than or equal to 10 and less than or equal to 20 reduce the **total** price by 5%

**You should reduce the prices in that EXACT order**

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 30  Students  Sunday | Total price: 266.73 |
| 40  Regular  Saturday | Total price: 800.00 |

## Print and Sum

Write a program to display numbers from given start to given end and their sum. All the numbers will be integers. On the first line you will receive the start, on the second the end.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5  10 | 5 6 7 8 9 10  Sum: 45 |
| 0  26 | 0 1 2 … 26  Sum: 351 |
| 50  60 | 50 51 52 53 54 55 56 57 58 59 60  Sum: 605 |

## Login

You will be given a string representing a username. The password will be that username reversed. Until you receive the correct password print on the console “**Incorrect password. Try again.**”. When you receive the correct password print “**User {username} logged in.**” However on the fourth try if the password is still not correct print “**User {username} blocked!**” and end the program.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Acer  login  go  let me in  recA | Incorrect password. Try again.  Incorrect password. Try again.  Incorrect password. Try again.  User Acer logged in. |
| momo  omom | User momo logged in. |
| sunny  rainy  cloudy  sunny  not sunny | Incorrect password. Try again.  Incorrect password. Try again.  Incorrect password. Try again.  User sunny blocked! |

## Strong Number

Write a program to check if a given number is a strong number or not. A number is strong if the sum of the Factorial of each digit is equal to the number. For example 145 is a strong number, because **1! + 4! + 5! = 145.** Print "**yes**" if the number is strong and “**no**” if the number is not strong.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 2 | yes |
| 3451 | no |
| 40585 | yes |

## Vending Machine

Until you receive “**Start**” you will be given different coins that are being inserted in the machine. You have to sum them in order to have the total money inserted. There is a problem though.

Your vending machine only works with **0.1**, **0.2**, **0.5, 1, and 2** coins. If someone tries to insert some other coins you have to display “**Cannot accept {money}**” and **not** add it to the total money.

On the next few lines until you receive “**End**” you will be given products to purchase. Your machine has however only “**Nuts**”, “**Water**”, “**Crisps**”, “**Soda**”, “**Coke**”. The prices are: **2.0**, **0.7**, **1.5**, **0.8**, **1.0** respectively. If the person tries to purchase a not existing product print “**Invalid product**”.

Be careful that the person may try to purchase a product they don’t have the money for. In that case print “**Sorry, not enough money**”. If the person purchases a product successfully print “**Purchased {product name}**”.

After the “**End**” command print the money that are left formatted to the second decimal point in the format “**Change: {money left}**”.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 1  1  0.5  0.6  Start  Coke  Soda  Crisps  End | Cannot accept 0.6  Purchased coke  Purchased soda  Sorry, not enough money  Change: 0.70 |

## Triangle of Numbers

Write a program, which receives a number – **n**, and prints a triangle from **1 to n** as in the examples.

### Constraints

* **n** will be in the interval [**1...20]**.

### Examples

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 3 | 1  2 2  3 3 3 | 5 | 1  2 2  3 3 3  4 4 4 4  5 5 5 5 5 | 6 | 1  2 2  3 3 3  4 4 4 4  5 5 5 5 5  6 6 6 6 6 6 |

## **\*Padawan Equipment**

Yoda is starting his newly created Jedi academy. So, he asked master Ivan Cho to **buy** the **needed equipment**. The number of **items** depends on **how many students will sign up**. The equipment for the Padawan contains **lightsabers, belts and robes**.

You will be given **the amount of money Ivan Cho has**, the **number of students** and the **prices of each item**. You have to help Ivan Cho **calculate** if the **money** he has is **enough to buy all of the equipment**, or how much more money he needs.   
Because the lightsabres sometimes brakes, Ivan Cho should **buy 10% more (taken from the students count)**, **rounded up** to the next integer. Also, every **sixth belt is free**.

### Input / Constraints

The input data should be read from the console. It will consist of **exactly 5 lines**:

* The **amount of money** Ivan Cho has – **floating-point number** in **range [0.00…1,000.00]**
* The **count of students – integer in range [0…100]**
* The **price of lightsabers** for a **single sabre – floating-point number** in **range [0.00…100.00]**
* The **price of robes** for a **single robe – floating-point number** in **range [0.00…100.00]**
* The **price of belts** for a **single** **belt – floating-point number** in **range [0.00…100.00]**

The **input data will always be valid**. **There is no need to check it explicitly**.

### Output

The output should be printed on the console.

* **If the calculated price of the equipment is less or equal to the money Ivan Cho has:**
  + "The money is enough - it would cost {the cost of the equipment}lv."
* **If the calculated price of the equipment is more than the money Ivan Cho has:**
  + "Ivan Cho will need {neededMoney}lv more."
* **All prices** must be **rounded to two digits after the decimal point.**

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 100  2  1.0  2.0  3.0 | The money is enough - it would cost 13.00lv. | Needed equipment for 2 padawans :  sabresPrice\*(studentsCount + 10%) + robesPrice \* (studentsCount) + beltsPrice\*(studentsCount-freeBelts)  1\*(3) + 2\*(2) + 3\*(2) = 13.00  13.00 <= 100 – the money will be enough. |
| **Input** | **Output** | **Comments** |
| 100  42  12.0  4.0  3.0 | Ivan Cho will need 737.00lv more. | Needed equipment for 42 padawans:  12\*47 + 4\*42 + 3\*35 = 837.00  837 > 100 – need 737.00 lv. more. |

*...May the force  
 be with you...*

## \*Rage Expenses

As a MOBA challenger player, Pesho has the bad habit to trash his PC when he loses a game and rage quits. His gaming setup consists of **headset, mouse, keyboard and display**. You will receive Pesho`s **lost games count**.

Every **second** lost game, Pesho trashes his **headset.**

Every **third** lost game, Pesho trashes his **mouse**.

When Pesho trashes **both** **his mouse and headset** in the **same** lost game, he also trashes his **keyboard**.

**Every** **second time, when he trashes his keyboard**, he also trashes his **display**.

You will receive the price of each item in his gaming setup. Calculate his rage expenses for renewing his gaming equipment.

### Input / Constraints

* On the first input line - **lost games count** – integer in the range **[0, 1000]**.
* On the second line – **headset price** - floating point number in range **[0, 1000]**.
* On the third line – **mouse price** - floating point number in range **[0, 1000]**.
* On the fourth line – **keyboard price** - floating point number in range **[0, 1000]**.
* On the fifth line – **display price** - floating point number in range **[0, 1000]**.

### Output

* As output you must print Pesho`s total expenses: **"Rage expenses: {expenses} lv."**
* Allowed working **time** / **memory**: **100ms** / **16MB**.

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comment** |
| 7  2  3  4  5 | Rage expenses: 16.00 lv. | Trashed headset -> 3 times  Trashed mouse -> 2 times  Trashed keyboard -> 1 time  Total: 6 + 6 + 4 = 16.00 lv; |
| 23  12.50  21.50  40  200 | Rage expenses: 608.00 lv. |  |

01 Task 1 //

using System;

namespace Lecture\_5\_Exersizes

{

class Program

{

static void Main(string[] args)

{

int age = int.Parse(Console.ReadLine());

string typeOfPerson = string.Empty;

if (age >= 0 && age <= 2)

{

typeOfPerson = "baby";

}

else if (age >= 3 && age <= 13)

{

typeOfPerson = "child";

}

else if (age >= 14 && age <= 19)

{

typeOfPerson = "teenager";

}

else if (age >= 20 && age <= 65)

{

typeOfPerson = "adult";

}

else if (age >= 66)

{

typeOfPerson = "elder";

}

else if (age < 0)

{

Console.WriteLine($"ages {age} only positive.");

}

Console.WriteLine(typeOfPerson);

}

}

}

/ Task / 2

using System;

namespace NumbersDevision

{

class Program

{

static void Main(string[] args)

{

int num = int.Parse(Console.ReadLine());

int bigestDevider = int.MinValue;

int devisibleNum = 0;

if (num % 2 != 0 && num % 3 != 0 && num % 6 != 0 && num % 7 != 0 && num % 10 != 0 )

{

Console.WriteLine("Not divisible");

}

else // the number devides itself at least one number

{

if (num % 2 == 0)

{

devisibleNum = 2;

if (devisibleNum > bigestDevider)

{

bigestDevider = devisibleNum;

}

}

if (num % 3 == 0)

{

devisibleNum = 3;

if (devisibleNum > bigestDevider)

{

bigestDevider = devisibleNum;

}

}

if (num % 6 == 0)

{

devisibleNum = 6;

if (devisibleNum > bigestDevider)

{

bigestDevider = devisibleNum;

}

}

if (num % 7 == 0)

{

devisibleNum = 7;

if (devisibleNum > bigestDevider)

{

bigestDevider = devisibleNum;

}

}

if (num % 10 == 0)

{

devisibleNum = 10;

if (devisibleNum > bigestDevider)

{

bigestDevider = devisibleNum;

}

}

Console.WriteLine($"The number is divisible by {bigestDevider}");

}

}

}

}

Task 2 Second way //

using System;

namespace NumbersDevision

{

class Program

{

static void Main(string[] args)

{

int num = int.Parse(Console.ReadLine());

int bigestDevider = int.MinValue;

int devisibleNum = 0;

// the number devides itself at least one number

if (num % 2 == 0)

{

devisibleNum = 2;

if (devisibleNum > bigestDevider)

{

bigestDevider = devisibleNum;

}

}

if (num % 3 == 0)

{

devisibleNum = 3;

if (devisibleNum > bigestDevider)

{

bigestDevider = devisibleNum;

}

}

if (num % 6 == 0)

{

devisibleNum = 6;

if (devisibleNum > bigestDevider)

{

bigestDevider = devisibleNum;

}

}

if (num % 7 == 0)

{

devisibleNum = 7;

if (devisibleNum > bigestDevider)

{

bigestDevider = devisibleNum;

}

}

if (num % 10 == 0)

{

devisibleNum = 10;

if (devisibleNum > bigestDevider)

{

bigestDevider = devisibleNum;

}

}

if (num % 2 != 0 && num % 3 != 0 && num % 6 != 0 && num % 7 != 0 && num % 10 != 0)

{

Console.WriteLine("Not divisible");

}

if (bigestDevider != int.MinValue)

{

Console.WriteLine($"The number is divisible by {bigestDevider}");

} // the number has at least one devider if bigestDiveder initial value

// is different than the initial value from the beggining! : then number devider // is found !

}

}

}

Task 3 //

using System;

namespace VacationChoosePrice

{

class Program

{

static void Main(string[] args)

{

int numGroup = int.Parse(Console.ReadLine());

string typeOfGroup = Console.ReadLine();

string dayOfWeek = Console.ReadLine();

double priceOfStay = 0;

if (dayOfWeek == "Friday")

{

if (typeOfGroup == "Students")

{

priceOfStay = 8.45 \* numGroup;

if (numGroup >= 30)

{

priceOfStay = 0.85 \* priceOfStay;

}

}

else if (typeOfGroup == "Business")

{

priceOfStay = 10.90 \* numGroup;

if (numGroup >= 100 )

{

priceOfStay -= (10 \* 10.90);

}

}

else if (typeOfGroup == "Regular")

{

priceOfStay = 15 \* numGroup;

if (numGroup >= 10 && numGroup <= 20)

{

priceOfStay = 0.95 \* priceOfStay;

}

}

}

else if (dayOfWeek == "Saturday")

{

if (typeOfGroup == "Students")

{

priceOfStay = 9.80 \* numGroup;

if (numGroup >= 30)

{

priceOfStay = 0.85 \* priceOfStay;

}

}

else if (typeOfGroup == "Business")

{

priceOfStay = 15.60 \* numGroup;

if (numGroup >= 100)

{

priceOfStay -= (10 \* 15.60);

}

}

else if (typeOfGroup == "Regular")

{

priceOfStay = 20 \* numGroup;

if (numGroup >= 10 && numGroup <= 20)

{

priceOfStay = 0.95 \* priceOfStay;

}

}

}

else if (dayOfWeek == "Sunday")

{

if (typeOfGroup == "Students")

{

priceOfStay = 10.46 \* numGroup;

if (numGroup >= 30)

{

priceOfStay = 0.85 \* priceOfStay;

}

}

else if (typeOfGroup == "Business")

{

priceOfStay = 16 \* numGroup;

if (numGroup >= 100)

{

priceOfStay -= (10 \* 16);

}

}

else if (typeOfGroup == "Regular")

{

priceOfStay = 22.50 \* numGroup;

if (numGroup >= 10 && numGroup <= 20)

{

priceOfStay = 0.95 \* priceOfStay;

}

}

}

Console.WriteLine($"Total price: {priceOfStay:F2}");

}

}

}

////

Task : 4

// using System;

namespace PrintSumNumbers

{

class Program

{

static void Main(string[] args)

{

int startNum = int.Parse(Console.ReadLine());

int endNum = int.Parse(Console.ReadLine());

int sum = 0;

for (int i = startNum; i <= endNum; i++)

{

Console.Write($"{i} ");

sum += i;

}

Console.WriteLine();

Console.WriteLine($"Sum: {sum}");

}

}

}

//

5 Task ;

LogiIn // give me 50 % true, because of Limit time !

using System;

namespace EnterCorrectPassword

{

class Program

{

static void Main(string[] args)

{

string username = Console.ReadLine();

//first string this is username we read to generate pass

char[] pass = new char[username.Length];

for (int i = 0, j = username.Length - 1; i < username.Length; i++, j--)

{

pass[i] = username[j]; //We initalize array with correct reveresed password

}

string reversed = string.Join("", pass); // Here we create the password to Log in!

while (true)

{

string correctPassword = Console.ReadLine();

// Here we call User to enter correct pass that we compare

// with genereated revered pass from reversed array

if (correctPassword == reversed)

{

Console.WriteLine($"User {username} logged in.");

break;

}

Console.WriteLine("Incorrect password. Try again.");

}

}

}

}

//

Task 6; //// 100\100 – а имам малка печатна грешка.

using System;

namespace StrongNumberFactoriel

{

class Program

{

static void Main(string[] args)

{

int inputNum = int.Parse(Console.ReadLine());

int sumFactorielInputNum = 0;

int originNum = inputNum;

while (inputNum % 10 != 0)

{

int digitNum = inputNum % 10;

int remDigit = digitNum;

int factorielRemovedDigit = 1;

for (int i = 1; i <= remDigit; i++) // calculate the factoriel of the currect digit

{

factorielRemovedDigit \*= i;

}

sumFactorielInputNum += factorielRemovedDigit; // add all current digits factoriel to each other to sum the total

inputNum = inputNum / 10; // remove the last number from the input number

}

if (sumFactorielInputNum == originNum)

{

Console.WriteLine("yes");

}

else

{

Console.WriteLine("no");

}

}

}

}

//

Here by the checks of the input values =40585 the program gives me a mistake -> no, (I have a mistake when debugged).

so number =40585- is not strong according to my solution,

But according to the task condition input value = 40585 is strong number-> therefore my solution is not fully correct .

In judge when I enter my wrong solution it passes 100 % the tests, so the test say true;

I believe my solution is not true , because when reaching the number 40 –I don’t get into the while cycle and calculations are stopped ! I never enter into the while cycle.

So my result is wrong at the end. So the test must have caught this ! .Probably it is a test bug.

Please check it.

// Task 6 –Correct Solution

using System;

namespace FactorielStrongNumber2

{

class Program

{

static void Main(string[] args)

{

int input = int.Parse(Console.ReadLine());

int number = input; //We create this number bcasue it will change its value during execution,not to change the input value !

int currDigit = 0; // We createt this variable where to store the curr digits of the number

int factorialSum = 0; // the sum of all digits factoriel of the input we store here !

while (number != 0)

{

currDigit = number % 10; // We take from the curr number the current digit

int currDigitFactorial = 1; // we set it to one bcasue we will use it for multiplication

for (int i = 1; i <= currDigit; i++) // we walk through the loop to calculate curr digit factoriel !

{

currDigitFactorial \*= i;

}

factorialSum += currDigitFactorial; // we add each factoriel to the total sum of factoriels of all digits

number = number / 10; // we remove the last digit from the number , becasue it is already used for calculations!

}

if (factorialSum == input) // at the end we check and print

{

Console.WriteLine("yes");

}

else

{

Console.WriteLine("no");

}

}

}

}

////

Task 7…// My Decision gives 100% currest results with the OutPut Results from the Task,

But in Judge it says 100% untrue/

using System;

namespace VendingMashine

{

class Program

{

static void Main(string[] args)

{

string insertedItem = Console.ReadLine();

double budget = 0;

while ( insertedItem != "Start")

{

double currCoins = double.Parse(insertedItem);

if ( currCoins == 0.1 || currCoins == 0.2 || currCoins == 0.5 || currCoins == 1 || currCoins == 2)

{

budget += currCoins;

insertedItem = Console.ReadLine();

}

else

{

Console.WriteLine($"Cannot accept {currCoins}");

break;

}

}

insertedItem = Console.ReadLine();

double nutsPrice = 2.0, waterPrice = 0.7, crispsPrice = 1.5, sodaPrice = 0.8, cokePrice = 1.0;

double productPrice = 0;

double totalProductscosts = 0;

if (insertedItem == "Start")

{

string productName = Console.ReadLine();

while (productName != "End")

{

if (productName == "Nuts")

{

productPrice = nutsPrice;

}

else if (productName == "Water")

{

productPrice = waterPrice;

}

else if (productName == "Crisps")

{

productPrice = crispsPrice;

}

else if (productName == "Soda")

{

productPrice = sodaPrice;

}

else if (productName == "Coke")

{

productPrice = cokePrice;

}

else

{

Console.WriteLine("Invalid product");

break;

}

if (budget >= productPrice)

{

Console.WriteLine($"Purchased {productName.ToLower()}");

budget -= productPrice;

}

else

{

Console.WriteLine("Sorry, not enough money");

}

productName = Console.ReadLine();

}

Console.WriteLine($"Change: {budget:F2}");

}

}

}

}

//

07/Task-> Second solution ! TeachersSolution !

using System;

namespace VendingMashineTeacher

{

class Program

{

static void Main(string[] args)

{

string inputLine = Console.ReadLine();

double budget = 0;

while (inputLine != "Start")

{

double currCoin = double.Parse(inputLine);

if (currCoin == 0.1 || currCoin == 0.2 || currCoin == 0.5 ||

currCoin == 1 || currCoin == 2)

{

budget += currCoin;

}

else

{

Console.WriteLine($"Cannot accept {currCoin}");

}

inputLine = Console.ReadLine();

}

inputLine = Console.ReadLine();

while (inputLine != "End")

{

double productPrice = 0;

if (inputLine == "Nuts")

{

productPrice = 2.0;

}

else if (inputLine == "Water")

{

productPrice = 0.7 ;

}

else if (inputLine == "Crisps")

{

productPrice = 1.5;

}

else if (inputLine == "Soda")

{

productPrice = 0.8;

}

else if (inputLine == "Coke")

{

productPrice = 1.0;

}

if (productPrice != 0)

{

if(budget >= productPrice)

{

Console.WriteLine($"Purchased {inputLine.ToLower()}");

budget -= productPrice;

}

else

{

Console.WriteLine("Sorry, not enough money");

}

}

else

{

Console.WriteLine("Invalid product");

}

inputLine = Console.ReadLine();

}

Console.WriteLine($"Change: {budget:F2}");

}

}

}

Task 8

using System;

namespace TriangleNumbers

{

class Program

{

static void Main(string[] args)

{

int num = int.Parse(Console.ReadLine());

for (int rows = 1; rows <= num ; rows++)

{

for (int cols = 0 ; cols < rows; cols++)

{

Console.Write(rows + " ");

}

Console.WriteLine();

}

}

}

}

Task/ 9 //

using System;

namespace PadawanEquipment

{

class Program

{

static void Main(string[] args)

{

double budget = double.Parse(Console.ReadLine());

int studensCount = int.Parse(Console.ReadLine());

double saberPrice = double.Parse(Console.ReadLine());

double robePrice = double.Parse(Console.ReadLine());

double beltPrice = double.Parse(Console.ReadLine());

double countSabers = Math.Ceiling(studensCount \* 1.1);

// every sixth belt is free ! minus from the result !

double totalCostsEquipment = 0;

if (studensCount >= 6) // counte belts = count Students-> one belt pro Student !

{

totalCostsEquipment = saberPrice \* countSabers +

robePrice \* studensCount +

beltPrice \* (studensCount - (studensCount / 6));

}

else

{

totalCostsEquipment = saberPrice \* countSabers +

robePrice \* studensCount +

beltPrice \* studensCount ;

}

if (totalCostsEquipment <= budget)

{

Console.WriteLine($"The money is enough - it would cost {totalCostsEquipment:F2}lv.");

}

else

{

Console.WriteLine($"Ivan Cho will need {(totalCostsEquipment - budget):F2}lv more.");

}

}

}

}

Task 9 Second way :

using System;

namespace PadawanEquipment

{

class Program

{

static void Main(string[] args)

{

double budget = double.Parse(Console.ReadLine());

int studensCount = int.Parse(Console.ReadLine());

double saberPrice = double.Parse(Console.ReadLine());

double robePrice = double.Parse(Console.ReadLine());

double beltPrice = double.Parse(Console.ReadLine());

double countSabers = Math.Ceiling(studensCount \* 1.1);

// every sixth belt is free ! minus from the result !

double totalCostsEquipment = totalCostsEquipment = saberPrice \* countSabers +

robePrice \* studensCount +

beltPrice \* studensCount;

if (studensCount >= 6) // count belts = count Students-> one belt pro Student !

{ // enters here for calvulation only students count is >= 6; then it reasignes the :double totalCostsEquipment

totalCostsEquipment = saberPrice \* countSabers +

robePrice \* studensCount +

beltPrice \* (studensCount - (studensCount / 6));

}

if (totalCostsEquipment <= budget)

{

Console.WriteLine($"The money is enough - it would cost {totalCostsEquipment:F2}lv.");

}

else

{

Console.WriteLine($"Ivan Cho will need {(totalCostsEquipment - budget):F2}lv more.");

}

}

}

}

/////

10 // Задача ;

using System;

namespace LostGamesCount

{

class Program

{

static void Main(string[] args)

{

int lostGamesCounter = int.Parse(Console.ReadLine());

double headsetPrice = double.Parse(Console.ReadLine()); // 2 LostG -> 1-HeadSet

double mousePrice = double.Parse(Console.ReadLine()); // 3 LostG -> Mouse

double keyboardPrice = double.Parse(Console.ReadLine()); // 6 LostG -> Haedset + Mouse + 1 Keyboard

double displayPrice = double.Parse(Console.ReadLine()); // 12 LostG -> HaedSet + Mouse + 1 Keyboard + 1 display

// double totalCosts = 0;

int brokenHeadsetCounter = lostGamesCounter / 2; // find the number of broken HeadSet

int brokenMouseCounter = lostGamesCounter / 3;

int brokenKeyboardCounter = lostGamesCounter / 6;

int brokenDisplayCounter = lostGamesCounter / 12;

double totalCosts = headsetPrice \* brokenHeadsetCounter + mousePrice \* brokenMouseCounter +

keyboardPrice \* brokenKeyboardCounter + displayPrice \* brokenDisplayCounter ;

Console.WriteLine($"Rage expenses: {totalCosts:F2} lv.");

}

}

}

//

2 Second way solving the task -10 ! heavier ! //the Teachers Solving !

using System;

namespace LostGamesCount

{

class Program

{

static void Main(string[] args)

{

int lostGamesCounter = int.Parse(Console.ReadLine());

double headsetPrice = double.Parse(Console.ReadLine()); // 2 LostG -> 1-HeadSet

double mousePrice = double.Parse(Console.ReadLine()); // 3 LostG -> Mouse

double keyboardPrice = double.Parse(Console.ReadLine()); // 6 LostG -> Haedset + Mouse + 1 Keyboard

double displayPrice = double.Parse(Console.ReadLine()); // 12 LostG -> HaedSet + Mouse + 1 Keyboard + 1 display

// double totalCosts = 0;

// First way of Solving the Task/

//int brokenHeadsetCounter = lostGamesCounter / 2; // find the number of broken HeadSet

//int brokenMouseCounter = lostGamesCounter / 3;

//int brokenKeyboardCounter = lostGamesCounter / 6;

//int brokenDisplayCounter = lostGamesCounter / 12;

//double totalCosts = headsetPrice \* brokenHeadsetCounter + mousePrice \* brokenMouseCounter +

// keyboardPrice \* brokenKeyboardCounter + displayPrice \* brokenDisplayCounter ;

//Console.WriteLine($"Rage expenses: {totalCosts:F2} lv.");

// Sec WAy of solving the task

int brokenHeadsetCounter = 0;

int brokenMouseCounter = 0;

int brokenKeyboardCounter = 0;

int brokenDisplayCounter = 0;

for (int i = 1; i <= lostGamesCounter; i++)

{

if (i % 2 == 0 )

{

brokenHeadsetCounter++;

}

if (i % 3 == 0)

{

brokenMouseCounter++;

}

if (i % 6 == 0)

{

brokenKeyboardCounter++;

}

if (i % 12 == 0)

{

brokenDisplayCounter++;

}

}

double totalCosts = headsetPrice \* brokenHeadsetCounter +

mousePrice \* brokenMouseCounter +

keyboardPrice \* brokenKeyboardCounter +

displayPrice \* brokenDisplayCounter ;

Console.WriteLine($"Rage expenses: {totalCosts:F2} lv.");

}

}

}