# Упражнения: По-сложни цикли

Тествайте решенията си в **Judge системата**: <https://judge.softuni.org/Contests/Practice/Index/3899>

Състояние на настоящия учебен материал:

A green and blue rectangle with white text

Description automatically generated

## Четни степени на 2

Да се напише програма, която чете число n, въведено от потребителя, и печата **четните степени на 2** **≤** **2n**:

* **20, 22, 24, 26, …, 2n.**

### Примери

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Вход** | **Изход** |  | **Вход** | **Изход** |  | **Вход** | **Изход** |  | **Вход** | **Изход** |  | **Вход** | **Изход** |
| 3 | 1  4 | 4 | 1  4  16 | 5 | 1  4  16 | 6 | 1  4  16  64 | 7 | 1  4  16  64 |

### Насоки

1. Прочетете от конзолата **числото** **n** –крайната степен, до която трябва да достигнете:

Text

Description automatically generated with medium confidence

1. Създайте **for-цикъл**, който започва от **0** (нулева степен) и достига до **n**. Стъпката трябва да е **+2**, защото трябва отпечатваме **само четните степени** (0, 2, 4, 6 и т.н.):

A picture containing logo

Description automatically generated

1. Във for-цикъла **изчислете резултата**, използвайки метода за степенуване **Math.Pow()**:



1. Отпечатайте **резултата** на конзолата:



## Лява и дясна сума

Да се напише програма, която чете **2\***n на брой цели числа, подадени от потребителя, и проверява дали **сумата на първите** n **числа** (**лява** сума) е равна на **сумата на вторите** n **числа** (**дясна** сума). При равенство печата "Yes, sum = " + **сумата**. Иначе печата " No, diff = " + **разликата**. Разликата се изчислява като **положително** число (по абсолютна стойност).

### Примери

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Вход** | **Изход** | **Коментар** |  | **Вход** | **Изход** | **Коментар** |
| **2**  10  90  60  40 | Yes, sum = 100 | 10+90 = 60+40 = 100 | **2**  90  9  50  50 | No, diff = 1 | 90+9 ≠ 50+50  Difference = |99-100| = 1 |

### Насоки

1. Прочетете от конзолата входните данни - **числото** **n**:



1. Създайте **две променливи**, които да съхраняват **лявата** и **дясната сума**:

A picture containing graphical user interface

Description automatically generated

1. Добавете **for-цикъл**, който започва от **1** и достига до **n \* 2**, защото трябва да имате **n** на брой числа **отляво** и **n** на брой числа **отдясно**, т.е. **2 пъти по n**:

A picture containing chart

Description automatically generated

1. На **всяка итерация** на for-цикъла прочетете **ново входно число** от конзолата:

A picture containing text

Description automatically generated

1. Проверете към **коя сума** спада съответното число – ако индексът е **по-малък или равен** на **n**, числото се добавя към **лявата сума**, а ако индексът е **по-голям** от **n**, числото се добавя към **дясната сума**:

Graphical user interface

Description automatically generated

1. **След цикъла** проверете дали **двете суми са равни** и отпечатайте съответния изход:

Graphical user interface, text

Description automatically generated

## Четна / нечетна сума

Да се напише програма, която получава число n на първия ред. След това чете n на брой цели числа, подадени от потребителя, и проверява дали **сумата от числата на четни позиции** е равна на **сумата от числата на нечетни позиции**. При равенство да се отпечатат **два реда**: "Yes" и на нов ред "Sum = " + **сумата**. Иначе да се отпечата "No" и на нов ред "Diff = " + **разликата**. Разликата се изчислява по **абсолютна стойност**.

### Примери

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** |  | **Вход** | **Изход** | **Обяснения** |  | **Вход** | **Изход** | **Обяснения** |
| 4  10  50  60  20 | Yes  Sum = 70 | 10+60 = 50+20 = 70 | 4  3  5  1  -2 | No  Diff = 1 | 3+1 ≠ 5-2  Diff = |4-3| = 1 | 3  5  8  1 | No  Diff = 2 | 5+1 ≠ 8  Diff = |6-8| = 2 |

### Насоки

1. Прочетете входните данни – **числото n**.
2. Създайте две променливи, които да съхраняват **сумата от числата на четните** и **сумата от числата на** **нечетните позиции**:

A picture containing graphical user interface

Description automatically generated

1. Създайте **for-цикъл**, който да взима **n** на брой числа, т.е. да **започва от първото** и да **достига до последното** (n):

A picture containing chart

Description automatically generated

1. За **всяка итерация** на **for-цикъла** прочетете от конзолата ново число **num**.
2. Проверете дали новото число е **четно** или **нечетно** и го **добавете към съответната сума**:

Text

Description automatically generated

1. **След цикъла** проверете дали **двете суми са равни** и отпечатайте **съответните съобщения**.

## Редица числа 2K+1

Напишете програма, която чете **число** n, въведено от потребителя,и отпечатва **всички числа ≤ n от редицата**: 1, 3, 7, 15, 31, …. Всяко следващо число се изчислява като **умножим** **предишното** с **2** и **добавим** **1**.

### Примери

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Вход** | **Изход** |  | **Вход** | **Изход** |  | **Вход** | **Изход** |  | **Вход** | **Изход** |
| 3 | 1  3 | 8 | 1  3  7 | 17 | 1  3  7  15 | 31 | 1  3  7  15  31 |

### Насоки

1. Прочетете от конзолата входните данни - **числото** **n**.
2. **Създайте** променлива, която ще представлява **брояч** и има **първоначална стойност 1**:

Logo

Description automatically generated

1. **Създайте while-цикъл**, който се повтаря докато броячъте **по-малък** **или равен** на числото, което сте прочели от конзолата:

Chart, scatter chart

Description automatically generated

1. При всяко повторение на цикъла **принтирайте стойността на брояча** и му **прибавяйте дадената стойност**:

Graphical user interface

Description automatically generated with medium confidence

## Баланс по сметка

Напишете програма, която пресмята **колко общо пари има в сметката**, след като направите определен **брой вноски**. На всеки ред ще получавате **сумата**, която трябва да внесете в сметката, до получаване на **команда "NoMoreMoney"** . При всяка получена сума на конзолата трябва да се извежда **"Increase: "** + **сумата**, закръглена до **втория знак след десетичната запетая.** След това **добавете сумата в сметката**. Ако получите число **по-малко от 0**, на конзолата трябва да се изведе **"Invalid operation!"** и **цикълът да приключи**. Когато цикълът приключи, трябва да се принтира **"Total: "** + **общата сума** в сметката, закръглена **до втория знак след десетичната запетая**.

### Примери

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Вход** | **Изход** |  | **Вход** | **Изход** |
| 5.51  69.42  100  NoMoreMoney | Increase: 5.51  Increase: 69.42  Increase: 100.00  Total: 174.93 | 120  45.55  -150 | Increase: 120.00  Increase: 45.55  Invalid operation!  Total: 165.55 |

### Насоки

1. Създайте променлива **totalSum**, която да съхранява **общата сума** в банковата сметка.
2. Създайте променлива **input**, която да съхранява **входните данни**, въведени на първия ред.
3. Създайте **while-цикъл**, който ще итерира до получаване на **команда "NoMoreMoney"**:

A picture containing logo

Description automatically generated

1. За **всяка итерация** на while-цикъла прочетете **сумата**, която трябва да се внесе:

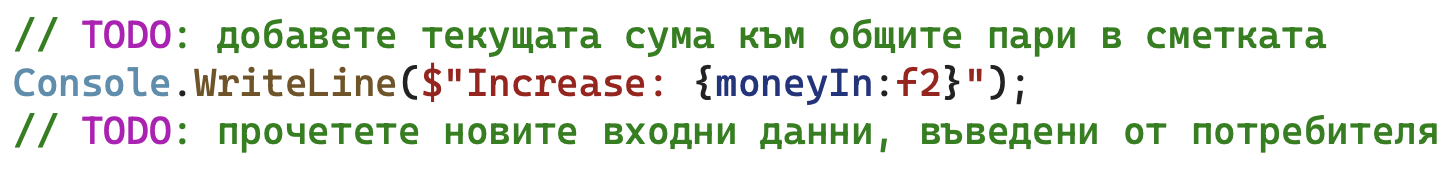


1. Проверете дали числото е **по-малко от 0**. В такъв случай отпечатайте **"Invalid operation!"** и **прекратете** **цикъла**:

Logo

Description automatically generated

1. Ако числото е валидно, **добавете сумата** към **сметката**.
2. Отпечатайте очакваното **съобщение за внесени пари**.
3. Прочетете **нови входни данни**, които потребителят въвежда:



## Завършванe

Напишете програма, която изчислява **средната оценка** на ученик от цялото му обучение. На първия ред ще получите **името на ученика**, а на всеки следващ ред - неговите **годишни оценки**. Ученикът преминава в следващия клас, **ако годишната му оценка е по-голяма или равна на 4.00**. **Ако ученикът бъде скъсан повече от един път, то той бива изключен и цикълът приключва,** като се отпечатва **името на ученика и в кой клас бива изключен**.

При успешно завършване на **12-ти** клас да се отпечата :

"{**име на ученика**} graduated. Average grade: {**средната оценка от цялото обучение**}"

**В случай че ученикът е изключен от училище, да се отпечата:**

"{**име на ученика**} has been excluded at {**класа, в който е бил изключен**} grade"

**Стойността трябва да бъде форматирана до втория знак** **след десетичната запетая**.

### Примери

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Вход** | **Изход** |  | **Вход** | **Изход** |
| Gosho  5  5.5  6  5.43  5.5  6  5.55  5  6  6  5.43  5 | Gosho graduated. Average grade: 5.53 | Mimi  5  6  5  6  5  6  6  2  3 | Mimi has been excluded at 8 grade |

### Насоки

1. Прочете от конзолата **името на ученика**.
2. Създайте променлива, която да съхранява **текущия клас** на ученика и има **начална стойност 1** (първи клас).
3. Създайте променлива, която да съхранява **сумата от оценките** на ученика и има **начална стойност 0**.
4. Създайте променлива, която да съхранява **броя на изпитите**, на които ученикът е **скъсан**.
5. Създайте **while-цикъл**, който итерира докато ученикът **завърши 12-ти клас**.
6. На **всяка итерация** на цикъла прочетете **нова оценка** от конзолата.
7. Ако оценката е **по-голяма или равна на 4.00**, я добавете към **сумата от оценки**.
8. Ако оценката е **по-малка от 4.00**, значи ученикът е **скъсан** и трябва да **инкрементирате** съответния брояч.
9. Ако ученикът е **скъсан повече от 1 път**, **прекратете** цикъла.
10. Ако ученикът е **завършил класа** успешно, **инкрементирайте** съответната променлива, за да премине в следващия клас.
11. **След цикъла** отпечатайте съответния изход в зависимост от това дали ученикът **е завършил 12-ти клас**, или **не**.

## Преместване

На осемнадесетия си рожден ден на Хосе взел решение, че ще се изнесе да живее на квартира. Опаковал багажа си в **кашони** и намерил подходяща обява за **апартамент под наем**. Той започва да пренася своя багаж **на части**, защото не може да пренесе целия наведнъж. Има **ограничено** **свободно пространство** в новото си жилище, където може да разположи вещите, така че мястото да бъде подходящо за живеене.

Напишете програма, която изчислява **свободния обем от жилището на Хосе, който остава след като пренесе багажа си.**

**Бележка**: Един кашон е с точни размери: 1m. x 1m. x 1m.

### Вход

Потребителят въвежда следните данни на отделни редове:

* **Широчина** на свободното пространство- цяло число в интервала [1...1000]
* **Дължина** на свободното пространство - цяло число в интервала [1...1000]
* **Височина** на свободното пространство- цяло число в интервала [1...1000]
* На следващите редоведо получаване на **команда "Done" -** **брой кашони**, които се пренасят в квартирата - цели числа в интервала [1...10000];

Програмата трябва да **приключи** прочитането на данни при **команда "Done"** или ако **свободното място свърши.**

### Изход

Да се **отпечата** на конзолата **един** от следните редове:

* Ако стигнете до командата **"Done"** и има още свободно място:

**"{брой свободни куб. метри} Cubic meters left."**

* Ако свободното място свърши преди да е дошла команда **"**Done**":**

**"No more free space! You need {брой недостигащи куб. метри} Cubic meters more."**

### Примери

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** |
| 10  10  2  20  20  20  20  122 | No more free space! You need 2 Cubic meters more. | 10 \* 10 \* 2 = **200 кубични метра.** 20 + 20 + 20 + 20 + 122 = **202** **кубични метра.** 200 - 202 = **2 недостигащи** кубични метра |
| 10  1  2  4  6  **Done** | 10 Cubic meters left. | 10 \* 1 \* 2 = **20 кубични метра.** 4 + 6 = **10** **кубични метра.** 20 - 10 = **10 кубични метра.** |

### Насоки

1. Прочетете входните данни – **широчина**, **дължина** и **височина** на свободното пространство.
2. Изчислете **обема** на свободното пространство.
3. Създайте променлива, която да съхранява **обема на кашоните**, които Хосе ще успее да премести в жилището си, с **начална стойност 0**.
4. Прочетете от конзолата **първата команда**, въведена от потребителя.
5. Създайте **while-цикъл**, който да итерира до достигане на **команда** "**Done**".
6. На **всяка итерация** на цикъла **добавете кашоните**, които е пренесъл в жилището, към **общия обем** на кашоните.

Ако **обемът на кашоните** стане **по-голям** от **свободния обем** в жилището, отпечатайте **"No more free space! You need {брой недостигащи куб. метри} Cubic meters more."** и **прекратете цикъла**.

1. Ако обемът на кашоните **не е надвишил** свободното пространство, прочетете **нови входни данни**.
2. Ако след изпълнението на цикъла има останало пространство, отпечатайте **"{брой свободни куб. метри} Cubic meters left."**.

## Сума от две числа

Напишете програма, която проверява **всички възможни комбинации** от **двойка числа** в интервала между **две дадени числа**. На изхода се отпечатва **коя поред е комбинацията**, чийто **сбор от числата** **е равен** на дадено **магическо число**. Ако няма **нито една комбинация**, отговаряща на условието, се отпечатва **съобщение, че не е намерено**.

### Вход

Входът се чете от конзолата и се състои от **три реда**:

* **Начало на интервала** – цяло число в интервала [1...999]
* **Край на интервала** – цяло число в интервала [по-голямо от първото число...1000]
* **Магическото число** – цяло число в интервала [1...10000]

### Изход

На конзолата трябва да се отпечата **един ред,** според резултата:

* Ако **е намерена** **комбинация**, чийто **сбор** на числата е **равен на магическото число**
  + "Combination N:{пореден номер} ({първото число} + {второ число} = {магическото число})"
* Ако **не е намерена** комбинация, отговаряща на условието
  + "{броят на всички комбинации} combinations - neither equals {магическото число}"

### Примери

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** | **Вход** | **Изход** |
| 1  10  5 | Combination N:4 (1 + 4 = 5) | Всички комбинации от две числа между 1 и 10 са:  1 1, 1 2, 1 3, **1 4**, 1 5, ... 2 1, 2 2, ... 4 9, 4 10, 5 1 ... 10 9, 10 10  Първата комбинация, чиито сбор на числата е равен на магическото число 5 е **четвъртата (1 и 4)** | 88  888  1000 | Combination N:20025 (112 + 888 = 1000) |
| **Вход** | **Изход** | **Обяснения** | **Вход** | **Изход** |
| 23  24  20 | 4 combinations - neither equals 20 | Всички комбинации от две числа между 23 и 24 са: 23 23, 23 24, 24 23, 24 24 (общо 4)  Няма двойки числа, чиито сбор е равен на магическото 20 | 88  888  2000 | 641601 combinations - neither equals 2000 |

## Сграда

Напишете програма, която **визуализира схематично етажите и помещенията в една сграда**. Да се изведат на конзолата **номерата на стаите** в сградата (в **низходящ ред**), като са изпълнени следните условия:

* На всеки **четен** етаж има **само офиси**
* На всеки **нечетен** етаж има **само апартаменти**
* Всеки **апартамент** се означава по следния начин : **А**{**номер на етажа**}{**номер на апартамента**}, номерата на апартаментите **започват от 0.**
* Всеки **офис** се означава по следния начин : **О**{**номер на етажа**}{**номер на офиса**}, номерата на офисите **също започват от 0.**
* На **последния етаж винаги има апартаменти** и те са по-големи от останалите, затова **пред номера им пише 'L', вместо 'А'.** Ако има **само един етаж,** то има само **големи апартаменти**.

От конзолата се прочитат **две цели числа - броят на етажите** и **броят на стаите за един етаж.**

### Примери

|  |  |  |  |
| --- | --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** | |
| 6  4 | L60 L61 L62 L63  A50 A51 A52 A53  O40 O41 O42 O43  A30 A31 A32 A33  O20 O21 O22 O23  A10 A11 A12 A13 | Имаме общо 6 етажа, с по 4 стаи на етаж. Нечетните етажи имат само апартаменти, а четните само офиси. | |
| **Вход** | **Изход** | **Вход** | **Изход** |
| 9  5 | L90 L91 L92 L93 L94  O80 O81 O82 O83 O84  A70 A71 A72 A73 A74  O60 O61 O62 O63 O64  A50 A51 A52 A53 A54  O40 O41 O42 O43 O44  A30 A31 A32 A33 A34  O20 O21 O22 O23 O24  A10 A11 A12 A13 A14 | 4  4 | L40 L41 L42 L43  A30 A31 A32 A33  O20 O21 O22 O23  A10 A11 A12 A13 |

## GUI App – Сграда

Реализирайте предходната задача като **Windows Forms GUI app**. Потребителският интерфейс може да изглежда така:

**Graphical user interface, text, application

Description automatically generated**

### Насоки

1. Създайте нов **Windows Forms проект** и му задайте **смислено име**, например "**BuildingApp**"
2. Променете **името** на формата: "**Form1**" 🡪 "**FormBuilding**"
3. Променете **заглавието** на формата "**Form 1**" 🡪 "**Building**"
4. Добавете необходимите **контроли**, примерно ето така:

Graphical user interface, application

Description automatically generated

Контролите трябва да са следните:

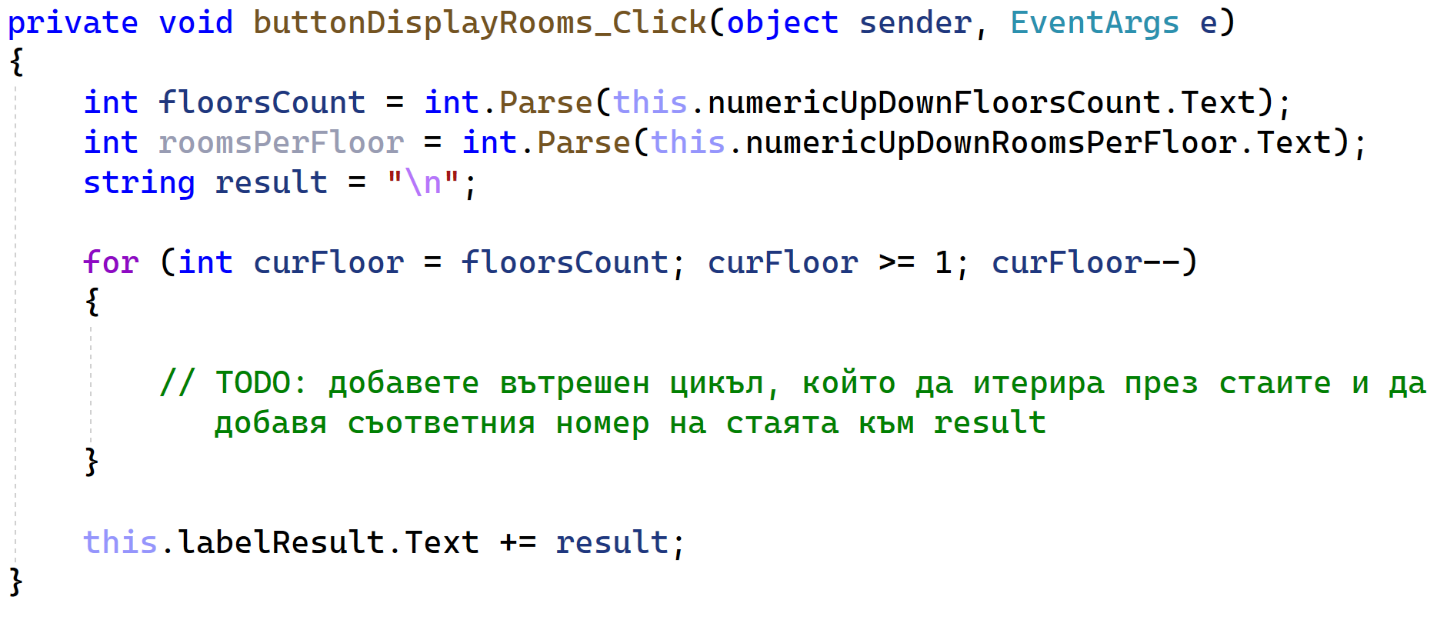
* **Три** надписа (**Label**):
  + "**labelFloorsCount**" с текст "**Floors count**: "
  + "**labelRoomsPerFloor**" с текст "**Rooms per floor:** "
  + "**labelResult**" с текст "**Result:** “
* **Две** числови полетa (**NumericUpDown**):
  + "**numericUpDownFloorsCount**" с максимална стойност **5**
  + "**numericUpDownRoomsPerFloor**" с максимална стойност **10**
* **Един** бутон (**Button**):
  + "**buttonDisplayRooms**" с текст "**Display rooms**"

1. Добавете **метод-обработчик** на бутона **buttonDisplayRooms** при събитието **Click** (чрез двоен клик в празното пространство):

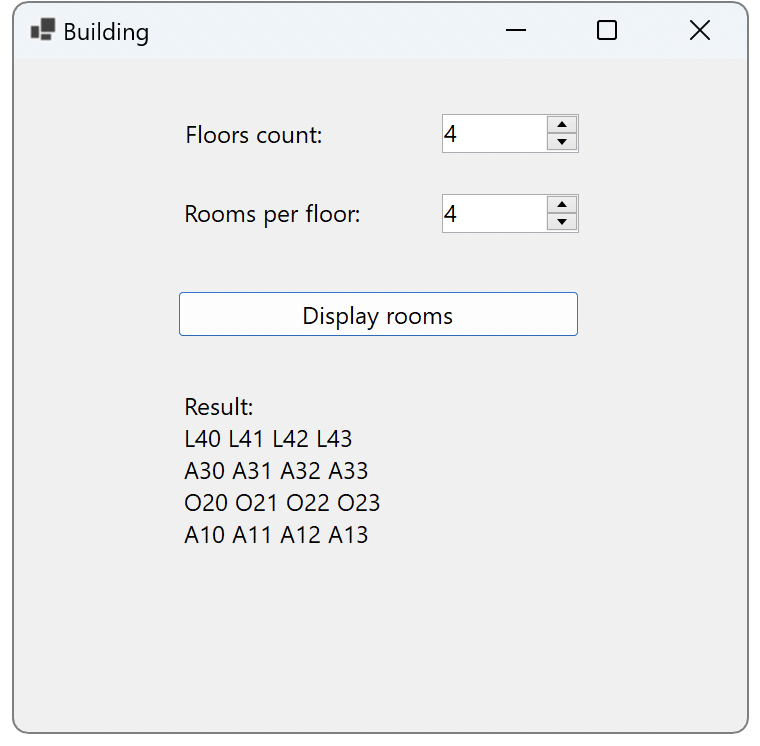
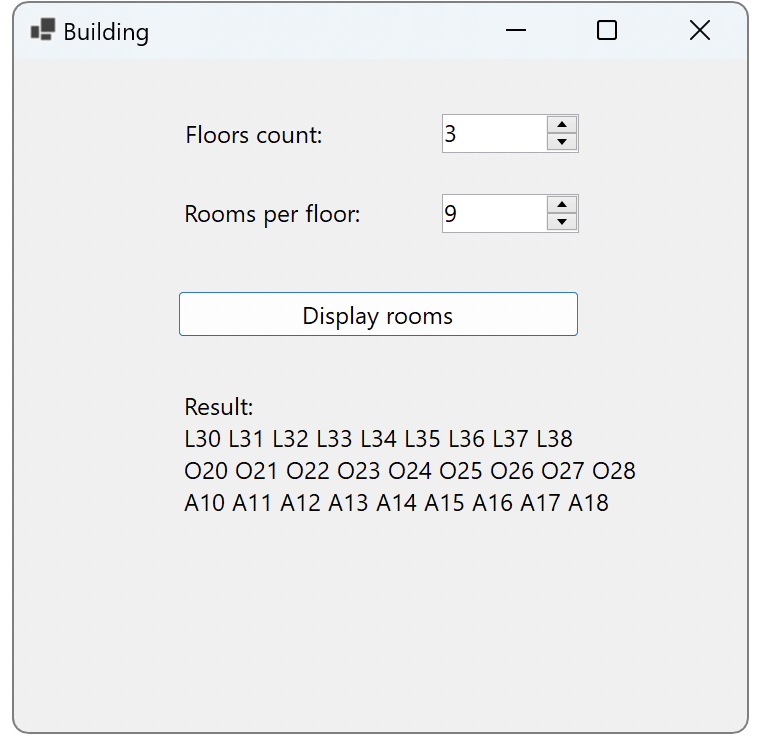
Graphical user interface, text, application, email

Description automatically generated

1. Добавете код в метода-обработчик, който да визуализира всички **етажи** и **стаи** на сградата в надписа **labelResult**:



1. Стартирайте приложението с **[Ctrl + F5]** и го тествайте:

**** ****

1. Можете ли да промените полето за резултат като използвате отделно многоредово текстово поле (TextBox.Multiline) и да използвате **monospaced шрифт**, така че визуализацията на сградата да бъде подредена вертикално по колони?

A picture containing text, font, screenshot, white

Description automatically generated

В настоящата имплементация стаите не са една под друга в колонки защото шрифтът не е подходящ.

## Заплата

**Шеф на компания забелязва, че все повече служители прекарват време в сайтове, които ги разсейват.**

**За да предотврати това, той въвежда изненадващи проверки на отворените табове на браузъра на служителите си. Според отворения сайт се налагат различни глоби:**

* **"Facebook" → 150 лв.**
* **"Instagram" → 100 лв.**
* **"Reddit" → 50 лв.**

**Напишете програма, която анализира списъка от отворените табове за даден служител и изчислява какви глоби да му бъдат наложени според правилата по-горе. Спрямо началната заплата на служителя и спрямо наложените глоби, да се изчисли остатъкът от заплатата, след глобите.**

### Вход

Първоначално от конзолата се четат **два реда:**

* **Брой отворени табове** в браузъра **n** - цяло число в интервала [1...10]
* **Заплата** - число в интервала [500...1500]

След това **n на брой пъти** се чете **име на уебсайт – текст**

### Изход

* **Ако по време на проверката заплатата стане по-малка или равна на 0 лева, на конзолата се изписва "You have lost your salary."** и програмата **приключва.**
* **В противен случай след проверката на конзолата се изписва остатъкът от заплатата (да се изпише като цяло число)**

### Примери

|  |  |  |  |
| --- | --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** | |
| 10  750  Facebook  Dev.bg  Instagram  Facebook  Reddit  Facebook  Facebook | You have lost your salary. | Има 10 отворени таба в браузъра.  Заплатата е 750  За първия таб -> Facebook глоба 150 лв.(750 – 150 = 600)  За втория таб -> Dev.bg не глобяват  За третия таб -> Instagram глоба 100 лв.(600 – 100 = 500)  За четвъртия таб -> Facebook глоба 150 лв.(500 – 150 = 350)  За петия таб -> Reddit глоба 50 лв. (350 – 50 = 300)  За шестия таб -> Facebook глоба 150 лв.(300 – 150 = 150)  За седмия таб -> Facebook глоба 150 лв.(150 – 150 = 0)  Заплатата е равна на 0, следователно се изписва съответният изход и програмата приключва. | |
| **Вход** | **Изход** | **Вход** | **Изход** |
| 3  500  Github.com  Stackoverflow.com  softuni.bg | 500 | 3  500  Facebook  Stackoverflow.com  softuni.bg | 350 |

## Еднакви суми на четни и нечетни позиции

Напишете програма, която чете от конзолата **две шестцифрени цели числа** в диапазона от **100000** до **300000**. Винаги **първото** въведено число ще бъде **по-малко от второто**. На конзолата да се отпечатат на **1 ред** **разделени с интервал** всички числа, които се намират **между двете** прочетени от конзолата числа, и отговарят на **следното** **условие**:

* **сумите** от цифрите на **четни** и **нечетни** позиции да са **равни**. Ако няма числа, отговарящи на условието, на конзолата **не се извежда резултат**.

### Примери

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** | | | |
| 100000  100050 | 100001 100012 100023 100034 100045 | Първото число, което генерираме е числото 100000. Сумата от цифрите на четни позиции (жълто) е 0+0+0=0. Сумата от цифрите на нечетни позиции (зелено) е 0+0+1=1. Тъй като двете суми са различни числото не се отпечатва.  Следващото, число е 100001. Сумата на четни позиции е 1+0+0=1, а на нечетни 0+0+1=1. Двете суми са равни и числото се отпечатва.  Следващото число за проверка е 100002. То не отговаря на условието и не се отпечатва.  ……  При числото 100045 сумата от четните позиции е 5+0+0=5, а на нечетни 4+0+1=5. Двете суми са равни числото се отпечатва. И т.н. | | | |
| **Вход** | **Изход** | **Вход** | **Изход** | **Вход** | **Изход** |
| 123456  124000 | 123464 123475 123486 123497 123530 123541 123552 123563 123574 123585 123596 123640 123651 123662 123673 123684 123695 123750 123761 123772 123783 123794 123860 123871 123882 123893 123970 123981 123992 | 299900  300000 | 299970 299981 299992 | 100115  100120 | *Няма изход* |