# Списъци – обработки

Тествайте задачите от тази тема в judge: [https://judge.softuni.bg/Contests/2657](https://judge.softuni.bg/Contests/2657/Списъци-обхождане)

## Списък от четни числа

Въведете **списък от цели числа** и изведете **четните** числа от списъка на един ред в конзолата. Елементите на списъка ще получите от единствен ред, разделени с интервали.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 3 **4 8** 5 7 5 **2** 1 | 4 8 2 |
| 1 **2 4** 3 | 2 4 |
| 7 **2 8** 3 5 9 7 3 | 2 8 |

## Списък от крайности

Въведете **списък от цели числа** и изведете тези от тях, които са равни на **минималния или максималния елемент**.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 5 4 8 5 7 8 2 1 | 1 8 8 |
| 1 1 1 | 1 1 1 |
| 4 2 8 3 5 9 2 3 | 2 2 9 |

## Максимална поредица еднакви числа

Въведете **списък от цели числа** и намерете **най-дългата поредица от еднакви елементи**. Ако съществуват няколко, отпечатайте **най-лявата**.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 3 4 4 **5 5 5** 2 2 | 5 5 5 |
| **7 7** 4 4 5 5 3 3 | 7 7 |
| 1 2 **3 3** | 3 3 |

### Подсказки

* Обходете позициите **p** отляво надясно и пазете **началото** и **дължината** на текущата поредица от еднакви числа приключваща с **p**.
* Също така пазете текущата най-добра (най-дълга) поредица (bestStart – позицията, на която започва, както и bestLength – нейната дължина) и я обновявайте след всяка стъпка

## Сума на обърнати числа

Напишете програма, която прочита поредица от цели числа, преобръща техните цифри и ги сумира.

### Примери

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Пояснения** |
| 123 234 12 | 774 | 321 + 432 + 21 = 774 |
| 12 12 34 84 66 12 | 220 | 21 + 21 + 43+ 48 + 66 + 21 = 220 |
| 120 1200 12000 | 63 | 21 + 21 + 21 = 63 |

## Премахни числото

Въведете **списък от цели числа** и премахнете всички срещания в списъка на последното число. Елементите на списъка ще получите от единствен ред, разделени с интервали.

**Примери**

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 3 4 **1** 5 **1** 5 2 **1** | 3 4 5 5 2 |
| 7 **3** 8 **3** 5 **3** 7 **3** | 7 8 5 7 |
| **2 2** 8 **2** 5 **2** 3 **2** | 8 5 3 |

**Подсказки**

* Извлечете стойността на последния елемент. Той се намира на индекс равен на **броя на елементите минус 1.** Броят на елементите може да разберете чрез **Count**
* Докато елементът съществува, премахвайте първото му срещане чрез **Remove**

## Изтриване на отрицателни елементи

Въведете **списък от цели числа, премахнете всички отрицателни числа** от него и го изведете на конзолата в **обратен ред**. В случай, че в списъка не са останали елементи, изведете “**empty**”.

**Примери**

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 10 -5 7 9 -33 50 | 50 9 7 10 |
| 7 -2 -10 1 | 1 7 |
| -1 -2 -3 | Empty |

**Подсказки**

* Създайте нов празен списък за получения като резултат списък
* Обходете въведения списък отзад напред. Проверете всеки елемент и добавете неотрицателните елементи към списъка за резултат
* Накрая, изведете списъка резултат на единствен ред, разделен с интервали.

## Сливане на списъци

Напишете програма, която **слива няколко списъка** от числа.

* Списъците се разделят от ‘**|**’.
* Стойностите се разделят от интервали (‘’, един или няколко)
* Подредете списъците **отзад** **напред**, а техните стойности **отляво** **надясно**.

**Примери**

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 1 2 3 |4 5 6 | 7 8 | 7 8 4 5 6 1 2 3 |
| 7 | 4 5|1 0| 2 5 |3 | 3 2 5 1 0 4 5 7 |
| 1| 4 5 6 7 | 8 9 | 8 9 4 5 6 7 1 |

**Подсказки**

* Създайте нов празен списък за резултатите.
* Отделете входа чрез ‘**|**’ така че да се получи списък от низове.
* Обходете получения списък отдясно наляво.
  + За всеки низ в списъка: отделете елементите му чрез знака за интервал
  + Всеки един елемент, който е непразен низ, трябва да бъде добавен към списъка с резултата
* Изведете списъка с резултата

## Бомбички

Напишете програма, която **въвежда поредица от числа** и **специално число - бомбичка** с определена **сила**. Вашата задача е да **детонирате всяко срещане на специалното число бомба** и според нейната сила **нейните съседи отляво и отдясно**. Детонациите се изпълняват отляво надясно и всички детонирани числа изчезват. Най-накрая изведете **сумата от оставащите елементи** в поредицата.

**Примери**

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Коментари** |
| 1 2 2 4 2 2 2 9  4 2 | 12 | Бомбичката е **4** със сила 2. След детонацията остават [1, 2, 9] със сума 12. |
| 1 4 4 2 8 9 1  9 3 | 5 | Бомбичката е **9** със сила 3. След детонацията оставаме с поредицата [1, 4], която има сума 5. Понеже 9 има само 1 съсед отдясно, ние го премахваме |
| 1 7 7 1 2 3  7 1 | 6 | Детонациите се изпълняват отляво надясно. Не можем да детонираме второто срещане на 7, понеже то вече е унищожено от първата детонация. Остават [1, 2, 3]. Тяхната сума е 6. |
| 1 1 2 1 1 1 2 1 1 1  2 1 | 4 | Оцветените числа изчезват в две последователни детонации. Оставащата предица е [1, 1, 1, 1]. Нейната сума е 4. |

## Сортиране на числа

Въведете **списък от цели числа** и го сортирайте.

**Примери**

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 8 2 7 3 | 2 <= 3 <= 7 <= 8 |
| 1 1 | 1 <= 1 |
| 2 4 -9 | -9 <= 2 <= 4 |
| 1 -0.5 | -0.5 <= 1 |

## Числа квадрати

Въведете **списък от цели числа** и **изведете всички числа квадрати** от списъка в **низходящ ред**. **Число квадрат** е цяло число, което е квадрат на друго цяло число. Например, 1, 4, 9, 16 са числа квадрати.

**Примери**

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 3 **16 4** 5 6 8 **9** | 16 9 4 |
| 12 **1 9 4 16** 8 **25 49 16** | 49 25 16 16 9 4 1 |

**Подсказки**

* За да разберете дали едно цяло число е “**число квадрат**”, проверете дали неговия корен квадратен е цяло число (такова че да няма дробна част):
  + **if (√num == (int)√num) …**
* За да подредите списъка от резултати в намаляващ ред използвайте сортиране с ламбда функция:
  + **squareNums.Sort((a, b) => b.CompareTo(a));**

## Брой на числа

Въведете **списък от цели числа** в интервала [0…1000] и **ги изведете в нарастващ ред** заедно с **броя на срещанията им**.

**Примери**

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 8 2 2 8 2 2 3 7 | 2 -> 4  3 -> 1  7 -> 1  8 -> 2 |
| 10 8 8 10 10 | 8 -> 2  10 -> 3 |

**Подсказки**

1. Въведете елементите в масива от цели числа **nums[]**. Например: {8, 2, 2, 8, 2, 2, 3, 7}.
2. Сортирайте **nums[]** в нарастващ ред: {2, 2, 2, 2, 3, 7, 8, 8}. Сега намерете всички поредици от едни и същи числа.
3. **Обходете** числата отляво надясно. Пребройте колко пъти се среща всяко число
   * Започнете с **count** = **1**.
   * Докато следващото число отдясно е **същото** като сегашното, **увеличавайте** **count** и продължете към следващото число.
   * Когато числото отдясно е **различно** (или няма друго число), **изведете** текущия елемент и неговия брой.
   * Продължете да обхождате от следващото число отдясно.