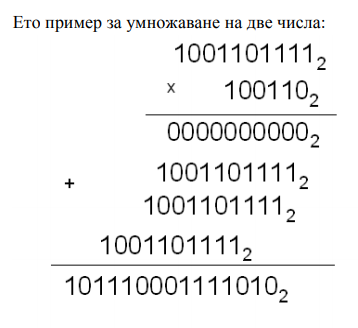
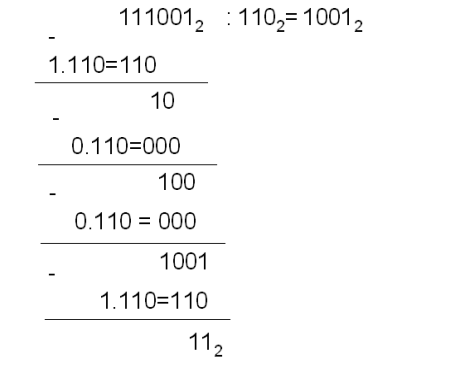
## C:\Users\Plamena\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Untitled.png

## Тук най-десните цифри (цифрите от първи разред – цифрите в първа позиция на двете събираеми) са съответно 1 и 0 и на тях съгласно таблицата за събиране отговаря цифрата 1. Следващите две цифри са 1 и 1 – пишем 0 и правим пренос наляво с една позиция, имаме “едно наум”. В третата позиция имаме цифрите 0 и 0, на които отговаря цифрата 0, но тук е пренесена една единица, следователно имаме съпоставяне на цифрата 0 с цифрата 1 - на тези цифри ще отговаря съгласно таблицата цифрата 1, която пишем в резултата като цифра от трети разред. В четвърта позиция имаме цифрите 1 и 0 – пишем 1, в петата позиция на цифрите 0 и 1 също отговаря 1. В шестата позиция се намират цифрите 1 и 1. На тях отговаря цифрата 0 и правим пренос на една единица наляво. Там първото събираемо има за последна цифра цифрата 1, а второто събираемо няма цифра, но за пълнота можем да си мислим (и пишем) цифрата 0 и на тях ще съответства цифрата 1, а на нея и пренесената единица ще отговаря 0 и ще имаме последен пренос, който направо отразяваме в резултата с най-лява цифра 1.

Умножение на двоични числа Умножаването на две числа в двоична бройна система се основава на умножаване на числата 0 и 1 и получаване на резултата посредством събиране с подходящо подреждане на позициите по същия начин, както в десетичната.



Деление на двоични числа Както знаем, да разделим естественото число а на естественото число b ще рече да намерим тяхното частно q и остатъка r, т.е. да намерим единствените цели числа q и r, за които a=bq+r, 0≤r. Тук r се нарича още остатък на а по модул b, а самото деление е известно и като “деление с остатък”. Когато числата a и b са записани в двоична бройна система, делението с остатък се извършва по същия начин, както при десетичната система – алгоритъмът не зависи от бройната система, но се осъществява технически в нея посредством междинни остатъци, към които приписваме отдясно по една цифра, започвайки отляво надясно. Това се вижда от следващия пример (с подробности за междинните резултати):



Частното от делението на 111001 с 110 е равно на 1001 а остатъкът е 11 . Следователно имаме равенството: 111001 =110 .1001 +11

|  |  |  |
| --- | --- | --- |
| **Decimal** | **Binary** | **Hex** |
| 0 | 0000 | 0 |
| 1 | 0001 | 1 |
| 2 | 0010 | 2 |
| 3 | 0011 | 3 |
| 4 | 0100 | 4 |
| 5 | 0101 | 5 |
| 6 | 0110 | 6 |
| 7 | 0111 | 7 |
| 8 | 1000 | 8 |
| 9 | 1001 | 9 |
| 10 | 1010 | A |
| 11 | 1011 | B |
| 12 | 1100 | C |
| 13 | 1101 | D |
| 14 | 1110 | E |
| 15 | 1111 | F |

**От Двоична в Шестнайсетична**

Ако искаме да сметнем числото 01101000101000101111 (2) до число с база (16), първо се разделя двоичното число на групи от по четири:

0110 1000 1010 0010 1111

Гледайки горната таблица заместваме:

0110 = 6

1000 = 8

1010 = A

0010 = 2

1111 = F

Когато се съберат се получава, че 01101000101000101111 (2) = 68A2F(16).

**От Шестнайсетична в Двоична**

Числото 68A2F(16) в двоична бройна система:

6 = 0110

8 = 1000

А = 1010

2 = 0010

F = 1111

Когато се съберат се получава, че = 68A2F(16) = 01101000101000101111 (2).

**От Десетична в Шестнайсетична**

Имаме числото 428591. Всяка цифра се разделя на 16 и се взима нейният остатък. Полученото от делението число делим отново на 16, дотогава докато се получи 0 за резултат. Получените от остатъка числа се вземат отзад напред (от последното деление до първото) и се получава числото в шестнайсетична бройна система.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 428591 | ÷ | 16 | = | 26786 | остатък | 15 |
| 26786 | ÷ | 16 | = | 1674 | остатък | 2 |
| 1674 | ÷ | 16 | = | 104 | остатък | 10 |
| 104 | ÷ | 16 | = | 6 | остатък | 8 |
| 6 | ÷ | 16 | = | 0 | остатък | 6 |

Гледаме остатъка отзад напред и получаваме, че 428591(10) = 68A2F(16).

**От Шестнайсетична в Десетична**

Имаме числото 68A2F(16). Взимат се цифрите отзад напред, всяка се умножава с 16[0..n]. Резултатът се събира.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| F | \* | 160 | = | 15 | \* | 1 | = | 15 |
| 2 | \* | 161 | = | 2 | \* | 16 | = | 32 |
| A | \* | 162 | = | 10 | \* | 256 | = | 2560 |
| 8 | \* | 163 | = | 8 | \* | 4096 | = | 32768 |
| 6 | \* | 164 | = | 6 | \* | 65536 | = | 393216 |

15 + 32 + 2560 + 32768 + 393216 = 428591

И получаваме, че 68A2F(16) = 428591(10).

## Задача 1 Преобразуване на числа от една бройна система в друга

Преобразувайте 1234 в двоична и шестнадесетична бройна системи.

Преобразувайте 1100101b в десетична и шестнадесетична бройна системи.

Преобразувайте ABChex в десетично и двоична бройна системи.

## Задача 2 Двоична аритметика

Преобразувайте числата в двоична бройна система и извършете действията в двоича бройна система. След това преобразувайте резултата в десетична бройна система

12+15=

9+15=

13\*5=

17\*3=

36/4=

81/9=

https://www.convertworld.com/bg/chisla/shestnadesetichna.html

## Задача 4 (с точност 12 знака)

Напишете програма, в която въвеждаме радиус r (реално число) и **извеждаме лицето** на кръг с точно **12 знака** след десетичната запетая. Използвайте тип данни с **подходяща точност** за съхранение на резултатите.

### Примери

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Вход** | **Изход** |  | **Вход** | **Изход** |
| 2.5 | 19.634954084936 |  | 1.2 | 4.523893421169 |

### Подсказки

Може да използвате тип double. Той има точност 15-16 знака.

За да изведете точно 12 знака след десетичната запетая, може да ползвате следния код:



## Задача 5 Точна сума на реални числа

Напишете програма, която въвежда n числа и изчислява и извежда тяхната **точна сума** (без закръгляне).

### Примери

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Вход** | **Изход** |  | **Вход** | **Изход** |
| 3  1000000000000000000  5  10 | 1000000000000000015 |  | 2  0.00000000003  333333333333.3 | 333333333333.30000000003 |

### Подсказки

Ако използвате типове като float или double, резултатът ще изгуби точността си. Също така данните може да се изведат с експоненциален запис. Може да използвате типа decimal, който съхранява реални числа с висока точност и по-малка загуба.

**Задача 6** **Сортиране**

От клавиатурата се въвежда масив от цели числа. Сортирайте го в низходя ред. Изведете резулата на един ред

**Примери**

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 1 5 -11 35 -3 | -11 -3 1 5 |
| 84 2 90 110 34 6 | 2 6 34 84 110 |

**Упътване**

Намерете най-малкия елемент от масива и разменете мястото му с първия

Повтаряйте горната стъпка с елементите от втория до последния и т.н

## Задача 7 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 |

### Подсказски

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**

## Задача 8 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.

## Задача 9 Променлив списък

Напишете програма, която въвежда **списък от цели числа** от **конзолата** и получава **команди**, които **манипулират** списъка. Вашата програма може да получава следните команди:

Delete {елемент} – изтрива всички елементи в списъка, които са равни на дадения елемент

Insert {елемент} {позиция} – вмъква елемент на дадената позиция

Програмата трябва да приключва, когато получи команда Odd или Even. Ако програмата получи Odd 🡺 извежда всички **нечетни** числа в списъка отделени с **единствен** интервал, иначе извеждаме по същия начин всички **четни** числа.

### Примери

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Вход** | **Изход** |  | **Вход** | **Изход** |
| 1 2 3 4 5 5 5 6  Delete 5  Insert 10 1  Delete 5  Odd | 1 3 |  | 20 12 4 319 21 31234 2 41 23 4  Insert 50 2  Insert 50 5  Delete 4  Even | 20 12 50 50 31234 2 |

## Задача 10 Цензора

Напишете програма, която приема като входни данни, една **дума** и **изречение**. Вашата програма трябва да **търси** **думата** в **изречението** и замени **всяка** **буква** от думата с **"\*"**. Вие трябва да направите това за **всяко** **срещане** на думата. Заменете **само** думите, които са **напълно еднакви** с **думата** на **първия** ред. **Обърнете** внимание, че трябва да се **замени** думата, дори ако тя е част от **друга** дума.

### Вход

Входът ще се състои от **два реда**:

На първия ред, ще бъде дума, която трябва да се цензурира.

На втори ред ще бъдат изречението, които трябва да се цензурира.

### Изход

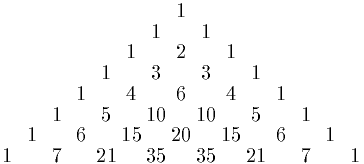
**Отпечатате** **изречението**, **след** **цензурирането**.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| money  Show me the money | Show me the \*\*\*\*\* |
| Doom  Doom and Gloom | \*\*\*\* and Gloom |
| Java  I love Java and JavaScript, but I hate Rxjava | I love \*\*\*\* and \*\*\*\*Script, but I hate Rxjava |

**Задача 11 Морски шах**

**Задача 12 Триъгълника на Паскал**



#### **Задача 13 Напишете програма, която намира най-честият брой в масив. Пример: {4, 1, 1, 4, 2, 3, 4, 4, 1, 2, 4, 9, 3} -> 4**

ОБЯСНЕНИЕ: С първите два for цикъла първо подреждаме самия масив от числа във възходящ ред. Стандартен алгоритъм за сортиране си е това, нищо сложно.

След което правим класическите три променливи, в които да отброяваме: броя на повторенията към момента, най-голямото повтериение и самото число, което е с най-голямото повторение.

Пускаме трети for цикъл, с който започваме да с сравняваме един по един елементите по елементите из масива и съответно ако има еднакви числа в поредица едно до друго if и else конструкцията в кода се грижат за това накрая конзолата да ни ги изпечата правилното решение на задачата.

#### **Задача 14** **Напишете програма, която намира в даден масив от цели числа последователност от дадена сума S (ако е налична). Пример: {4, 3, 1, 4, 2, 5, 8}, S = 11 -> {4, 2, 5}**

ОБЯСНЕНИЕ: Тук търсим в масив последователна поредица от числа, чиято сума да е равна на числото S. Правим си три променливи startIndex, endIndex, CurrSum.

С два for цикъла обхождаме масива събирайки отляво-надясно елементите му. В момента, в който открием елементи, чиято сума да удовлетворява if условието (а именно - временната CurrSum да бъде равна на sumS)

влизаме в третия for цикъл, който ни отпечатва индексите (startIndex e нейното начало, а endIndex нейния край) на търсената поредица от числа. Ако не намерим такава поредица от елементи - изписваме, че в масива няма такава поредица.

**Задача 15 Напишете програма, която чете правоъгълна матрица с размер N x M ( 3 x 3), и намерете максимална сума от нейните елементи.**

**Задача 16 Имате масив от стрингове. Напишете метод, който сортира масива по дължината на неговите елементи (броя на символите, които ги съставят).**