Задачи

Увод в програмирането

22 октомври 2018 г.

Comments

За всички фунции които пишете добавете коментари.
Опитайте се да напишете коментарите и декларицията на фунцията преди да напишете нейната имплементация (дефиниция). По този начин ще имате добра представа какво метода би трябвало да прави и как да се използва.

Използвайте този формат :

```
/**
The first sentence of each doc comment should be a summary sentence,
containing a concise but complete description of the method.

param param_name_1 - short description of the first parameter, including allowed and
not allowed values
param param_name_2 - ...
return - short descrption of the return value (if any)
*/
```

Например

```
/**
Reverses the given number. For example 123 -> 321

param input_number - integer positive number.
return - the reversed number. If it is "input_number" is negative assertion error will
be thrown.
*/
int reverse(int input_number) {}
```

Правила

- Използвайте функции. main() метода трябва да е максимално кратък.
- Пишете коментари по темплейта отгоре.
- Опитайте се да спазвате “Command Query Separation” където е възможно - т.е - отделете (command) фунции които имат страничен ефект (например принтират нещо от екрана) от (query) функции които връщат резултат (изчисляват нещо).

Задачи

1. *“Number series”*

Да се състави програма, чрез която се въвежда естествено число N и се извежда стойността на сумата от следните прости дроби: 1 / 2 + 1 / 6 + 1 / 12 + 1/20 + 1/30 + 1/42 + 1/56 + ...

Пример: N = 1 Изход: 0.500; N=2 Изход: 0.667; N=250 Изход: 0.996

2. *“Perfect Numbers Sum”*
   Съвършено число (Perfect number) се нарича естествено число, което е точна сума от своите по-малки делители (т.е. различни от самото число). Например най-малкото такова число е 6 - делителите му са 1, 2 и 3 и е изпълнено свойството: 1 + 2 + 3 = 6.
   Намерета сумата на всички съвършени число по-малки от N, където N е входен параметър.

   Пример: N=100 ; Изход: 34

3. *“Matrix with 0 diagonal”*
   Напишете програма в C ++, за да създадете квадратна матрица с 0 надолу по основния диагонал, 1 в записите точно над и под основния диагонал, 2 над и под това и т.н.
   Пример: N=4
   0 1 2 3
   1 0 1 2
   2 1 0 1
   3 2 1 0

4. *“Numbers”*
   a. Да се напише функция, която намира сумата на квадратните корени на всички положителни нечетни числа в даден затворен интервал [**a**; **b**].

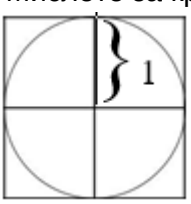
   b. Да се напише функция, която при вход цяло положително число (в десетична бройна система), проверява колко единици се съдържат в двоичния му запис.

5. “Digit Encounters”
   Да се напише функция, която проверява дали дадено цяло число съдържа всяка цифра в записа си:
   a. Само веднъж.

   b. Точно n пъти.

6. *“Password Generator”*
   Направете генератор на пароли. Бъдете креативни с това как генерирате пароли - силните пароли имат комбинация от малки букви, главни букви, цифри и символи. Паролите трябва да са произволни, като генерират нова парола всеки път, когато потребителят поиска нова парола. Структурирайте си кода с фунции по подходящ начин. Може да използвате rand() фунцията наготово
   (Hint: ASCII)

7. *“Calculating Pi”*
   Използвайки метода "Монте Карло" - т.е. рандомизирано симулиране - можем да изчислим с добра точност стойноста на PI.
   Мислете за кръг с радиус 1, и очертан около него квадрат:



Представете си, че това е мишена за стрелба и че вие хвърляте дартс в него на случаен принцип. При достатъчно хвърлени дартс съотношението на дартс в кръга към общия брой дартсове, които се

хвърлят,
трябва да бъде съотношението между площта на окръжността (A) (Pi*R^2=PI*1=PI) и площта на квадрата (4). С другуи думи ако имаме X дартца във окръжноста от общо N опита тогава X / N = (S-circle / S-square) = PI / 4 . Със 5 или 10 милиона опита трябва да е достатъчно да се получи добро приближение на Pi (3.14) . (Hint: use rand)