# **Упражнения:** *Функции и стойности*

## **Зад. 1** *Създайте функция*

Дефинирайте функция, която удвоява стойност.

### **Пример**

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 5 | 10 |

### **Подсказки**

1. Дефинирайте функция doubleVal, приемаща 1 параметър val и връща като резултат val + val

## **Зад. 2** *Проверка за четно число*

Дефинирайте функция, която проверява дали дадено цяло число е четно или нечетно.

### **Пример**

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 5 | False |
| 10 | True |

### **Подсказки**

1. Дефинирайте функцията isEven, приемаща 1 параметър val и връщаща като резултат True, ако числото е четно и False, ако не е
2. Разгледайте как работи вградената в Haskell функция mod и я използвайте в тялото на вашата функция

## **Зад. 3** *Най-голямото от три числа*

Дефинирайте функция, която приема като параметри 3 числа и връща най-голямото от тях.

### **Пример**

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 5 10 15 | 15 |
| 15 24 11 | 24 |

### **Подсказки**

1. Дефинирайте функция biggestOf3 приемаща 3 параметъра и връщаща най-големия от тях като резултат
2. Проверете дали първият параметър a е по-голям от вторият параметър b
   1. Ако е проверете дали първият параметър a е по-голям и от третият параметър c
      1. Ако е върнете a
      2. Ако не е върнет е c
   2. Ако a не е по-голям от b проверете дали b е по-голям от c
      1. Ако е върнете b
      2. Ако не е върнете c

## **Зад. 4** *Функция, изпълняваща друга функция*

Дефинирайте две функции, едната от които да приема като параметър число и да прибавя 1 към него. Нека другата приема като параметър функция, както и число, с което да се извика получената функция.

### **Пример**

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| add1 5 | 6 |
| remove1 7 | 6 |

### **Подсказки**

1. Дефинирайте функции, които главната функция ще извиква
   1. Например функция add1 приемаща 1 параметър - число и връщаща като резултат числото + 1
   2. Функция remove1 приемаща 1 параметър - число и връщаща като резултат числото - 1
2. Дефинирайте функция execute приемаща като параметри функция както и число, с което да извика приетата функция
   1. Като резултат върнете приетата функция с подаден параметър полученото число

## **Зад. 5** *Факториел*

Дефинирайте функция, която приема като параметър число n и връща като резултат n факториел

### **Пример**

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 5 | 120 |
| 10 | 3628800 |

### **Подсказки**

1. Дефинирайте функция factorial приемаща 1 параметър n
2. При n = 1 или n = 0 върнете като резултат 1 (дъно на рекурсията)
3. В противен случай върнете n умножено по-резултата получен при извикване на същата функция за n - 1

## **Зад. 6** *Редицата на Фибоначи*

Дефинирайте функция, която приема като параметър число n и връща като резултат n-тото число от редицата на Фибоначи. Редицата на Фибоначи започва от 1 и всяко следващо число е равно на сбора от предишните две - например второто число от редицата е равно на 1 + 0 (за нулево число от редицата се приема 0), третото число от редицата е равно на второто + първото (1 + 1) = 2 и тн.

### **Пример**

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 10 | 55 |
| 21 | 10946 |

### **Подсказки**

1. Дефинирайте функция fibonacci приемаща един параметър n и връщаща n-тото число от редицата като резултат
2. При n = 1 или n = 2 върнете като резултат 1 (дъно на рекурсията)
3. В противен случай за n върнете като полученият резултат от извикването на същата функция за n - 1 + резултат от извикването на същата функция за n - 2 (двете предишни числа от редицата на фибоначи)