# **Exercise: Recursion**

1. **Triangle**

Implement the following behavior through a Haskell function.

**Example**

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5 | \*\*\*\*\*  \*\*\*\*  \*\*\*  \*\*  \*  #  ##  ###  ####  ##### |
| 1 | \*  # |
| 4 | \*\*\*\*  \*\*\*  \*\*  \*  #  ##  ###  #### |

1. **Sum of first N Prime Numbers**

Implement a Haskell function which receives N and returns the **sum** of the first **N Prime numbers**. For this task let’s say prime numbers start from 2.

Prime Numbers: [read here](https://en.wikipedia.org/wiki/Prime_number).

**Example**

|  |  |
| --- | --- |
| **Input** | **Output** |
| 2 | 5 |
| 20 | 639 |
| 50 | 5117 |

**Note**

The cases above don’t represent an output to the Terminal. This is the result, returned from the function. In other words, your function should receive an array and return an array.

1. **Reverse Array**

Create a Haskell function, which receives an array and reverses it.

**Example**

|  |  |
| --- | --- |
| **Input** | **Output** |
| [1, 2, 3, 4, 5] | [5, 4, 3, 2, 1] |
| [1] | [1] |
| [] | [] |

**Note**

The cases above don’t represent an output to the Terminal. This is the result, returned from the function. In other words, your function should receive an array and return an array.

1. **Split String**

Create a Haskell function, which receives a string and a delimiter and returns an array which represents the string split by the delimiter into parts.

**Example**

|  |  |
| --- | --- |
| **Input** | **Output** |
| "qwe.rty.uio" "." | ["qwe", "rty", "uio"] |
| "asd" "" | ["a", "s", "d"] |
| "blahrow" "row" | ["blah", ""] |

**Note**

The cases above don’t represent an output to the Terminal. This is the result, returned from the function. In other words, your function should receive 2 strings and return an array.

1. **Read Until**

Implement a program with Haskell, which receives series of commands, reads until the command “**end**” is entered, and prints an array of the entered commands (excluding the “**end**” command)

**Example**

|  |  |
| --- | --- |
| **Input** | **Output** |
| Hello  World  !  end | ["Hello", "World", "!"] |
| Test  Test  Test  END  End  end | ["Test", "Test", "Test", "END", "End"] |

**Note**

The cases above don’t represent an output to the Terminal. This is the result, returned from the function. The **input, however**, is **entered** by the **user** through the **Terminal**.