Problem 1. Write the following functions:

read_integer : the function receives a message that will be prompt to the user who needs to enter an integer number. Then, it returns the number given by the user.

biggest : given two numbers, returns the biggest one.

Then, write a main program, to ask the user two numbers and print out the biggest one.

Problem 2. Using **docstrings**, introduce a good documentation to the functions from the previous exercise:

For the function `read_integer`:

```
Help text for the function  in module main :
read_integer()

This function returns an integer value requested
to the user with a suitable message using standard
input.
```

For the function `biggest`:

```
Help on function biggest in module main :
biggest(number1, number2)

Given two numbers, returns the biggest.
```

Problem 3. Write a main program to find the biggest of three numbers but restricting yourself to use the previous functions only.

```
Give me an integer number: 45
Give me another integer number: 32
Give me the last integer number: 46
The biggest is 46
```

Problem 4. Write a function that, given a year, returns true if it is a leap year; false otherwise. Then, write a main program checking every single year from 1 A.C. until now; print a suitable message for a year being or not leap.

Problem 5. Write a function that, given a student's mark (number), returns "F" [0,5), "E" [5,6), "D" [6,7), "C" [7,8), "B" [8,9), "A" [9,10) or "A+" (=10). Write a main program that asks the user to type a number in the range [0-10] and then prints the corresponding mark but do not return it.

Problem 6. Write a function that returns the nutritional status (or body mass index BMI) of a person according to the Table 1. The function will take two parameters weight (in kilos) and height (in metres).

Classification	**BMI (Kg/m2)**
Underweight	< 18.50
Normal	[18.50, 25.00)
Overweight	≥ 25.00
Obese	≥ 30.00

Table 1: BMI values for each label.

Comment the function with docstrings. Then, write main program that asks the user to enter weight and height (2 decimals) and prints the nutritional status.

Problem 7. Write a function to get the distance from a point $< x, y >$ (with x and y its coordinates) to the origin ($< 0, 0 >$). Then, write a main program to ask the user to enter the cartesian coordinates of a point in a plane and print the distance from that point to the origin. If you need help to use the math module, go to Python online documentation website: `https://docs.python.org/3/library/math.html`.

Problem 8. Using the previous function, write another function that returns the length of a circle whose center is the origin and passes through a point in the plane. To determine the length, you can use the value of π imported from the math package (`math.pi`). Write a program that prompts for the Cartesian coordinates of the point and displays the length of the circle centered in the origin and passing through that point.

Problem 9. Write a function that, given a positive integer number, returns its semi-factorial. Write a main program to test it. The semi-factorial of n is computed as $n \times (n-2) \times (n-4) \ldots$ until 2 or 1, depending on the nature of the number: even or odd.

Problem 10. Write a function that, given a positive integer number, calculates the sum of all its *proper divisors*. Then write a main program that asks the user to enter a positive integer and determines if the number is a *perfect number*. Then it prints a message. A *proper divisor* of n is any integer that divides n and that is different than n. A number n is *perfect number* if the sum of all its proper divisors equals the number n.

Problem 11. Write a program to print all the perfect numbers in the range [1,limit], where limit is a positive integer typed by the user.

Problem 12. Write a function, **isprime(...)**, that returns `True` if a given integer is prime and `False` otherwise. Then, write a program that asks the user to enter a positive integer and displays whether it is not prime. The program will keep asking numbers until the number is 0 or negative.

Problem 13. Write a function to check whether a number is in a given range. Then write a main program that asks the user to enter 3 positive integers, the number, the beginning of the range and the end of the range. Call the function and print a message.

Problem 14. Using the previous function, write a program that asks for two ranges (4 numbers in total) and print all the numbers in the intersection of these two ranges.

Problem 15. Write a function, to calculates the *n*th term in the following series:

$$a_k = \begin{cases} b & \texttt{if k equals 0} \\ c \times a_{k-1} + d & \texttt{k is higher than 0} \end{cases} \tag{1}$$

Then, write a main program that asks the user to enter values for b (first term in the series), c and d, as well as the term to calculate (positive integer). Finally, the program must print the value.

Problem 16. Write a function, that, given two positive integers n and b (b must be in the range [2, 16]) calculates the representation of the n in the numerical system with base b, representing it as a numerical string. Write a program that asks the user to enter a positive integer and prints it in base: 2, 8, 10 and 16.

Problem 17. Consider the function `line(ch, n)` below that returns a string with n characters ch.

```
def line(ch, n)
   """Returns a string with n characters ch"""
   return n * ch
```

1. Write a function `square(ch, n)` that prints a square with n lines and n columns full of characters ch. So `square('*',3)` must output:

   ```
   ***
   ***
   ***
   ```

2. Write a main program that asks character and number and then print the squared in the way described before.

Problem 18. Write a main program, similar to the previous one, to print just the outline of the square. For example, print `outlineSquare('*', 3)`:

```
******
*    *
******
```

Problem 19. Write a function called `fibonacci(n)` that return the nth term in the Fibonacci series. Then write a main program to test it.

$$f(n) = \begin{cases} 1 & \texttt{n equals 1 or n equals 2} \\ f(n-1) + f(n-2) & \texttt{n is higher than 2} \end{cases} \tag{2}$$

Problem 20. Using the previous `fibonacci(n)` function, write a main program that asks the user to enter a positive integer, m, and prints another series calculated summing the first k terms in the fibonacci series where k = 0, 1, 2,..., m. Example: m = 10 $\rightarrow$ 1 2 4 7 12 20 33 54 88 143 232

Problem 21. Write a shutting down function:

First, def a function, `shut_down`, that takes one paramter `s`. If `s` is equal to `"yes"`, it should print `"Shutting down"` and terminate the execution of the program. If `s` is equal to `"no"`, the function should print `"Shutdown aborted"`. Finally, if `s` is anything other than those inputs, the function should print `"Sorry"`. After that, write a program that uses this function.

Problem 22. Let's use functions to calculate your trip's costs:

Define a function called `hotel_cost` with one parameter `nights`. The hotel costs $140 per night. Define another function called `plane_ride_cost` that takes a string, `city`, as input. The function should return a different price depending on the location. Below are the valid destinations and their corresponding round-trip prices.

```
"Tokio": 183
"New York": 220
"London": 222
"Cabañaquinta": 475
```

Define a third function called `rental_car_cost` with an parameter called `days`. Calculate the cost of renting the car: Every day you rent the car costs $40. If you rent the car for 7 or more days, you get $50 off your total. Alternatively, if you rent the car for 3 or more days, you get $20 off your total. You cannot get both of the above discounts. Finally, define a function called `trip_cost` that takes three arguments, `city`, `days` and `spending_money` and calculates the sum of all the costs. Finally write a program that asks for all the depending variables and print the final cost.