Test 1: conditional statements, loops, functions

Quantitative Economics, Fall 2025

October 17, 2025

General notes

This test is designed to assess your understanding of fundamental programming concepts. It serves as a checkpoint to help you determine whether you have a solid grasp of the basics or if you need further practice. When writing code, you should follow **Julia** syntax; however, we will not be overly strict about small mistakes. The main goal is to demonstrate your understanding of the *logic* behind the code, not perfect memorization of syntax (although there are many syntax hints in this problem set—please make use of them!). Remember to close all conditional statements, loops, and functions with `end`; without this closure, the particular object is not properly defined. If you have any questions, don't hesitate to ask.

Note: No collaboration and no internet/AI use are allowed for this test. Complete it entirely on your own.

Your Name:

Conditional statements

Write a short Julia program that decides what a person should do with their free time based on the weather conditions.

Your program should:

- Print `"Go for a walk!"` if the temperature is greater than 15 and less[1] than 30, and[2] it is **not** raining.
- Print `"Go to a beach!"` if the temperature is greater than or equal to 30 and it is **not** raining.
- Print `"Stay inside!"` for all other cases.

[1] In Julia, the condition `a > x > b` will be true whenever `x` is greater than `b` and less than `a`.

[2] You will need some of these logical operators:

1. `&&` is "AND"
2. `||` is "OR"
3. `!` is "NOT"

Start by creating two variables:

```
temp = 25           #variable for the temperature
raining = false     #variable that is true whenever it rains
```

Note that you do not need to write any function here, just a conditional statement.

Loops

Write a short Julia program that multiplies all the numbers from 1 to 6. The goal is to define a variable `product` which, after the loop is executed, will be equal to:

$$\texttt{product} = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 \cdot 6$$

How to do it?

1. Define a variable `product` and set its initial value[3].
2. Use a `for` loop to go through all the numbers from 1 to 6[4].
3. Inside the loop, update the value of `product` in a way analogous to how we updated `my_sum` when calculating the sum[5].
4. After the loop finishes, print the final value of `product`.

Note that you do not need to write any function here, just a loop.

[3] When we were summing numbers using a loop, we initialized the variable `my_sum = 0`. Similarly, the variable `product` should also be initialized — but not with `0`. What value makes sense here?

[4] Remember that in Julia you can define a range over which a loop will execute using `1:1:6`, which means starting at 1, increasing by 1, until 6.

[5] Previously, when summing, we wrote `my_sum = my_sum + i`. Think carefully about what operation would play a similar role when finding a product.

Functions

Define a function `quadratic_roots(a, b, c)` that returns the real roots of a quadratic equation:

$$f(x) = ax^2 + bx + c$$

Note that the roots do not depend on x, so the function `quadratic_roots(a, b, c)` should take only a, b, and c as inputs. **NOTE:** In this problem, we assume that the discriminant (often called `delta`) is always greater than 0, so there will always be two real roots.

How to do it?

1. Start by defining a function. Recall that in Julia, a function can be defined using the following syntax:

```
function my_function(x)
    # some operations on the argument x
    return # return relevant object
end
```

   Define your function in a similar way, except that it should take the arguments `(a, b, c)`.

2. Inside the function, calculate the discriminant (often called `delta`[6]):

$$\Delta = b^2 - 4ac$$

3. Inside the function, define x_1 and x_2 in the following way:

$$x_1 = \frac{-b + \sqrt{\Delta}}{2a} \qquad x_2 = \frac{-b - \sqrt{\Delta}}{2a}$$

   In Julia, you can use the built-in square root function:

```
sqrt(delta)
```

4. Return x_1, x_2.

[6] Recall that in Julia, the mathematical expression x^2 is written as x^2.