Question 1 The Collatz sequence starting at a positive integer n is generated by repeating the following rule until the value 1 is reached: if the current value is even, divide it by 2; if it is odd, replace it by $3 \cdot \text{value} + 1$. Implement the Julia function `collatzsteps(n)` which returns the number of steps needed to reach 1 starting from `n`. Use a `while` loop.

For example, the sequence for $n = 6$ is $6, 3, 10, 5, 16, 8, 4, 2, 1$, which takes 8 steps. A successful test for your implementation would be:

```
julia> using Test
julia> @test collatzsteps(6) == 8
Test Passed
julia> @test collatzsteps(1) == 0
Test Passed
```

Question 2 Consider the following Julia code, run as a script:

```
x = 10

function g()
  x = 1
  for i in 1:3
    x += i
  end
  return x
end

println(g(), " ", x)
```

(a) What is printed?

(b) Which one line would you insert into `g` so that the output becomes `7 7`? Where would you insert it?

Question 3 Consider the Julia code

```
A = [1 2 3; 4 5 6; 7 8 9]
B = A[2:3, [1,3]]
C = A[:, 2]
D = A .* A'
E = A[end, :]'
```

(a) What is the value of `B`?

(b) What is the value and the type of `C`?

(c) What is the value of `D[3,2]`?

Question 4 Consider the Julia code

```
a = [1, 2, 3]
b = a
c = copy(a)
b[1] = 100
push!(c, 4)

f(v) = (v[end] = -1; v)
d = f(a)
```

What are the values of `a`, `b`, `c`, and `d` after running this code? What does the expression `d === a` evaluate to? Explain in one sentence.

Question 5 Let `s = "Julia is fun"`. State the result of each of the following expressions.

(a) `split(s)`

(b) `uppercase(s[1:5])`

(c) `replace(s, "u" => "U")`

(d) `"$(length(s)) chars: $(s[end-2:end])"`

Question 6 (a) What does the following Unix command do? Explain the role of each of its two parts and of the symbol between them.

```
ls *.jl | tail
```

Question 7 (a) Consider the following sorting algorithm implemented as function `insertionsort!(v)` which sorts the vector `v` in ascending order in place using the insertion sort algorithm, and returns `v`.

```
function insertionsort!(v)
  for i in 2:length(v)
    key = v[i]
    j = i - 1
    while j >= 1 && v[j] > key
      v[j+1] = v[j]
      j -= 1
    end
    v[j+1] = key
  end
  return v
end

julia> @test insertionsort!([5, 2, 4, 6, 1, 3]) == [1, 2, 3, 4, 5, 6]
Test Passed
```

(b) What is the worst-case time complexity of this implementation in terms of the length n of `v`? Which input is a worst case?

(c) What is the best-case time complexity and for which input does it occur?

Question 8 State the time complexity in Big-O notation of each of the following functions in terms of n (briefly justify your answer).

```
function g1(n)
  s = 0
  for i in 1:n, j in 1:i
    s += 1
  end
  return s
end

function g2(n)
  s = 0
  i = n
  while i >= 1
    s += 1
    i = div(i, 2)
  end
  return s
end

```

Question 9 Consider the following results in the Julia REPL.

```
julia> typemax(Int8) + Int8(1)
-128

julia> 2^64
0

julia> 0.1 + 0.2 == 0.3
false
```

(a) Explain the first result.

(b) Explain the second result and state how to compute 2^{64} correctly.

(c) Explain the third result and state how to compare the two numbers in a meaningful way.

Question 10 The derivative of a function f can be approximated by the forward difference

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}.$$

(a) Implement the Julia function `centraldiff(f, x, h)` which approximates $f'(x)$ using the central difference $\frac{f(x+h)-f(x-h)}{2h}$.

(b) The absolute error of the forward difference for $f = \sin$ at $x = 1$ was computed for several values of h:

h	10^{-1}	10^{-4}	10^{-8}	10^{-12}	10^{-16}
error	$4.3 \cdot 10^{-2}$	$4.2 \cdot 10^{-5}$	$3.0 \cdot 10^{-9}$	$4.3 \cdot 10^{-5}$	$5.4 \cdot 10^{-1}$

Explain why the error does not keep decreasing as h becomes smaller.

Question 11 (a) Implement the Julia function `euler(f, y0, t0, T, n)` which solves the initial value problem $y'(t) = f(t, y(t))$, $y(t_0) = y_0$ on the interval $[t_0, T]$ using n steps of the explicit Euler method with constant step size, and which returns the vector of length $n + 1$ containing the approximations of y at the grid points $t_0, t_0 + h, \ldots, T$.

A successful test for your implementation would be:

```
julia> @test euler((t, y) -> -y, 1.0, 0.0, 1.0, 2) == [1.0, 0.5, 0.25]
Test Passed
```

(b) Using $n = 100$ steps for the test problem above yields `0.3660` for the last component, the exact value is $e^{-1} \approx 0.3679$. What do you expect for the error if you use $n = 200$ steps instead? Justify your answer.

Question 12 (a) Implement a parametric `struct Point{T<:Real}` with two fields `x` and `y` of type `T`, and overload the addition operator `+` for two points (of arbitrary type parameters, adding coordinates).

A successful test for your implementation would be:

```
julia> Point(1, 2) + Point(0.5, 1.5)
Point{Float64}(1.5, 3.5)
```

(b) With your definition, you evaluate:

```
julia> Point(1, 2.5)
ERROR: MethodError: no method matching Point(::Int64, ::Float64)
The type ‘Point‘ exists, but no method is defined for this combination of argument types

Closest candidates are:
  Point(::T, ::T) where T<:Real
   @ Main REPL[23]:2

Stacktrace:
 [1] top-level scope
   @ REPL[28]:1
```

Explain in one sentence the reason for this error message.

(c) Add a method (a constructor) to the code which makes this call return `Point{Float64}(1.0, 2.5)`.

Question 13 Evaluate the following Julia expressions (each part is worth 1 mark; state `true`/`false` or the resulting type, respectively).

(a) `Int64 <: Integer`

(b) `Vector{Int} <: Vector{Number}`

(c) `Vector{Int} <: AbstractVector{<:Number}`

(d) `isconcretetype(Number)`

(e) `typeof([1, 2.0])`

(f) `supertype(Float64)`

Question 14 (a) Implement the Julia function `mygcd(a, b)` which computes the greatest common divisor of two non-negative integers using Euclid's algorithm as a recursive function.

(b) Write down the arguments of all recursive calls when evaluating `mygcd(252, 105)`.

Question 15 A polynomial $p(x) = c_1 + c_2x + \cdots + c_{n+1}x^n$ is represented by the vector of its coefficients `p = [c1, c2, ..., c(n+1)]`.

(a) Implement the function `horner(p, x)` which evaluates $p(x)$ using Horner's scheme, i.e. $p(x) = c_1 + x(c_2 + x(c_3 + \dots))$, with a single loop.

A successful test for your implementation would be:

```
julia> @test horner([1, 2, 3], 2) == 17
Test Passed
```

(b) How many multiplications does Horner's scheme need for a polynomial of degree n? How many multiplications would a naive evaluation need, in which every power x^k is computed from scratch (by $k-1$ multiplications) and then multiplied by c_{k+1}?

(c) Use Euclid's algorithm for polynomials to compute a greatest common divisor of $x^2 - 1$ and $x^2 + 2x + 1$. Show the steps.

Question 16 (a) Implement the Julia function `estimatepi(n)` which estimates π by a Monte Carlo simulation: draw `n` points uniformly distributed in the unit square $[0,1]^2$ and count the proportion that fall inside the quarter circle $x^2 + y^2 \leq 1$.

A successful test for your implementation would be:

```
julia> @test abs(estimatepi(10^6) - pi) < 0.01
Test Passed
```

(b) How can you make the output of the simulation reproducible?

Question 17 (a) The exponential distribution with rate $\lambda > 0$ has the cumulative distribution function $F(x) = 1 - e^{-\lambda x}$ for $x \geq 0$. Derive the inverse F^{-1} and explain how to use it, together with a uniform random number, to generate an exponentially distributed random number (inverse transform sampling).

(b) Implement the function `exprand(lam)` which returns one exponentially distributed random number with rate `lam` using only `rand()` as the source of randomness. What do you expect for the output of `mean([exprand(2.0) for _ in 1:10^6])`?

Question 18 A single server queue with first-in-first-out service is simulated with a discrete event simulation. Customers arrive at the times 1, 2, and 3, and the service of each customer takes exactly 2.5 time units. The server starts serving a customer as soon as the customer arrives or the previous customer has left, whichever happens later.

(a) List all events (arrival or departure of a customer) in the order in which they are processed by the simulation, together with the event time and the number of customers in the system immediately after the event.

(b) Which data structure is typically used to store the future events in a simulation engine, and why is it preferable to a simple unsorted vector?

Question 19 Consider the two functions

```
function f1(n)
  s = 0
  for i in 1:n
    s += i / 2
  end
  return s
end

function f2(n)
  s = 0.0
  for i in 1:n
    s += i / 2
  end
  return s
end
```

(a) What do `f1(3)` and `f2(3)` return?

(b) One of the functions is type-unstable. Which one, and why?

(c) How can you detect this problem using Julia tools, and how do you fix `f1`?

Question 20 Consider the `DataFrame`

```
julia> df = DataFrame(name=["A", "B", "C", "D"],
                      group=["x", "y", "x", "y"],
                      score=[10, 20, 30, 40])
 Row | name    group   score
     | String  String  Int64
-----+----------------------
   1 | A       x          10
   2 | B       y          20
   3 | C       x          30
   4 | D       y          40
```

(with `using DataFrames, Statistics`).

(a) What is the result of `combine(groupby(df, :group), :score => mean => :avg)`?

(b) Which rows are returned by `df[df.score .> 15, :]`?

(c) Add a column `rel` to `df` which contains each score divided by the maximal score.