❓ Question 1

When you run `np.spacing(1e7)`, you see `1.862645149230957e-09` as a result.

1. If you run `1e7 == 1e7+np.spacing(1e7)`, what will you see as a result? Why?
2. If you run `1e7 == 1e7+np.spacing(1e7)/3`, what will you see as a result? Why?

❓ Question 2

Please convert the following floating point number to IEEE754 representation (by double precision):
-10.125

❓ Question 3

Please convert the following IEEE754 representation to decimal (base 10) number (in double precision):
1 10000000011 10100000000.....000

❓ Question 4

Given the vector X = (x1, x2, x3), please explain and distinguish L1 and L2 norms.

❓ Question 5

What does "ill-conditioned matrix" mean? Please explain it in terms of numerical errors.

❓ Question 6

How do you check if a matrix has its corresponding inverse matrix? You may show it with an example.

❓ Question 7

The following Python code written in Ipython is given:

```
import numpy as np
np.spacing(1e15)
print(1e15 + (0.125/3))  # Output: 1000000000000000.0
```

1. Please explain what the code does.
2. Why does Python return 1e15 after the addition?

❓ Question 8

What is the largest number which is smaller than 7.1 in IEEE754 representation (in single precision)?

❓ Question 9

Please explain round-off error with an example.

❓ Question 10

Please write a class called `BankAccount` with the following specifications:

- Attributes: `__balance`, `__accountOnwer`, `__accountType`
- `__init__()` initializes the attributes
- Mutators: `set_balance()`, `set_accountOnwer()`, `set_accountType()`
- Accessors: `get_balance()`, `get_accountOnwer()`, `get_accountType()`

❓ Question 11

Which of the following will create an object, `worker_joey`, of the Worker class?

- [ ] `def__init__(worker_joey):`
- [ ] `class worker_joey:`
- [x] `worker_joey = Worker()`
- [ ] `worker_joey.Worker`

❓ Question 12

Which is the first line needed when creating a class named Worker?

- [ ] `def__init__(self):`
- [x] `class Worker:`
- [ ] `import random`
- [ ] `def worker_pay(self):`

? Question 13

What is the relationship called in which one object is a specialized version of another object?

- [ ] parent-child
- [ ] node-to-node
- [x] is a
- [ ] class-subclass

? Question 14

All instances of a class share the same values of the data attributes in the class.

- [ ] True
- [x] False

? Question 15

A class can be thought of as a blueprint that can be used to create an object.

- [x] True
- [ ] False

? Question 16

New attributes and methods may be added to a subclass.

- [x] True
- [ ] False

? Question 17

A value-returning function is like a simple function except that when it finishes it returns a value back to the part of the program that called it.

- [x] True
- [ ] False

? Question 18

A function definition specifies what a function does and causes the function to execute.

- [ ] True
- [x] False

? Question 19

One reason not to use global variables is that it makes a program hard to debug.

- [x] True
- [ ] False

? Question 20

What will be displayed after the following code is executed?

```
def pass_it(x, y):
    z = x * y
    result = get_result(z)
    return(result)

def get_result(number):
    z = number + 2
    return(z)

num1 = 3
num2 = 4
answer = pass_it(num1, num2)
print(answer)
```

- [ ] 12
- [ ] 9
- [x] 14
- [ ] Nothing, this code contains a syntax error.

? Question 21

In the following function, which of the parameters are keyword-only parameters?

```
def show_values(a, b, *, c, d):
    print(a, b, c, d)
```

- [ ] a and b
- [x] c and d
- [ ] a, b, and *
- [ ] a, b, c, and d

? Question 22

A set of statements that belong together as a group and contribute to the function definition is known as a

- [ ] header
- [x] block
- [ ] return
- [ ] parameter

? Question 23

A problem can normally be solved with recursion if it can be broken down into smaller problems that are identical in structure to the overall problem.

- [x] True
- [ ] False

? Question 24

Recursive algorithms are always more concise and efficient than iterative algorithms.

- [ ] True
- [x] False

? Question 25

Each time a function is called in a recursive solution, the system incurs overhead that is not incurred with a loop.

- [x] True
- [ ] False

❓ Question 26

Which would be the base case in a recursive solution to the problem of finding the factorial of a number?

- [x] n = 0
- [ ] n = 1
- [ ] n > 0
- [ ] The factorial of a number cannot be solved with recursion.

❓ Question 27

A problem can be solved with recursion if it can be broken down into __ problems.

- [x] smaller
- [ ] one-line
- [ ] manageable
- [ ] modular

❓ Question 28

In a recursive solution, if the problem cannot be solved now, then a recursive function reduces it to a smaller but similar problem and

- [ ] exits
- [ ] returns to the main function
- [ ] returns to the calling function
- [x] calls itself to solve the smaller problem