Java 2 Final Exam Questions

1. The granddaddy of everything in Java is said to be.
    a. Class
    b. Main method
    c. Node
    d. An Object
2. Which class defined `toString()` function which can be overridden by every sub-class.
    a. Object class
    b. String class
    c. ArrayList class
    d. Student class
3. The ArrayList<> can store any type including user-defined types. This is because the ArrayList class defined a type parameter. All this is possible because of__
    a. Object Oriented Programming
    b. Functional Programming
    c. Generic Programming
    d. None of the above
4. Enums are used to define__ types for variables.
    a. Dynamic
    b. Predefine
    c. Static
    d. Node of the above
5. Study the code snippets below to answer the following questions(5, 6, and 7).

```java
class Car {
    String name;
    String engine;
    int seats;
    int wheels;
}
```

With Object-Oriented Programming, we can define our type by creating a class that represents a real-world entity. For example, the `Car` class refers to what types?

   a. User define type

   b. String data type

   c. Object

   d. Constructor

6. The four variables defined at the class level are also called__ variable.

   a. Car

   b. Class

   c. Object

   d. Instance

7. The `Car` class does not explicitly define a constructor, but when creating a car object, Java will create a default construct under the hook. What is this constructor called?

   a. No-args constructor

   b. Super constructor

   c. Class constructor

   d. Object constructor

8. A function/sub-routine is a species of code that executes whenever called or invoked.

   a. true

   b. false

9. A stack data structure that stores items in a__ order.

   a. last in last out

   b. last in first out

   c. first in first out

   d. first in last out

10. What data structure does the stack utilize to store items internally?

    a. object

    b. list

    c. array

    d. trees

11. A data structure that stores items/elements using an index base is called.

a. an array

b. linked list

c. trees

d. all of the above

12. Which operation in LinkedList does not require changing its structure.

    a. insertion

    b. deletion

    c. displaying

    d. none of the above

13. Study the algorithm below and figure out its time and space complexity(questions; 13 and 14).

```
int[] temp = new int[this.data.length * 2];
for (int i = 0; i < data.length - 1; i++) {
            temp[i] = data[i];
      }
```

    The above code doubles an Array. What is the space complexity of the algorithm?

    a. O(1)

    b. O(n)

    c. both a and b

    d. none of the above

14. What is the time complexity of traversing through the data array?

    a. constant time O(1)

    b. The time is relative to the number of items in the data array, O(N)

    c. time-complexity not applicable

    d. none of the above

15. The below snippet is a `Node` class it has a value of type `int` and a next pointer pointing to a next Node.

```
private class Node {
        int value;
        Node next;
}
```

    What type of a LinkedList?

a. singly

b. doubly

c. circular

d. a and b

16. All the rest are principles of Object Oriented Programming except.

a. implements

b. abstraction

c. encapsulation

d. polymorphism

17. A recursive function is a function that calls itself, repetitively until a base case is reached.

a. true

b. false

c. both a and b

d. none the above

18. A data structure where insertion happens at the end and removal happens at the front.

a. Set

b. HashMap

c. Stack

d. Queue

19. In a LinkedList, the next pointer(s) are referential variables stored in the stack memory that point to an actual object stored in the heap memory.

a. false

b. true

20. Suppose `end = 3;` `val` will be inserted at which index.

```
data[end++] = val;
```

a. at index 2

b. at index 3

c. at index 4

d. none of the above

21. Study the code below.

```
static int sum(int a, int b){
        return a + b;
    }

static double sum(double a, int b){
    return a + b;
}
```

What programming technique allows you to write two or more functions/methods with the same but different method signatures?

a. method overridden

b. method overloading

c. recursion

d. generic programming

22. Study the code below and choose the right option.

```
public void insert(int val) {
        if (isFull()) {
            throw new IndexOutOfBoundsException("Queue is 
full cannot add item.");
        }

        data[end++] = val;
        end = end % data.length;
        size++;
    }
```

What will happen when the condition evaluates to true?

a. runtime error

b. compile time error

c. both a and b

d. none of the above

23. If a class extends a class it becomes__

a. super class

b. sub class

c. an object

d. a constructor

24. Objects are stored in the heap memory while their references are stored in__ memory.

    a. virtual

    b. stack

    c. array

    d. static

25. Below is a `CircularQueue` with two constructors. One is a `no-args` constructor and the other defines an argument. When creating a circular queue object it will be mandatory to pass an argument as the size of the internal data array.

```
public class CircularQueue {
    /*
    ....
    */

    public CircularQueue() {
        data = new int[DEFAULT_SIZE];
    }

    public CircularQueue(int size) {
        data = new int[size];
    }
}
```

    a. yes

    b. no

26. When working with data structures and performing certain operations will result in a change in the structure of the data structure. A `peek` operation on a Stack will result in a change in its structure.

```
public Task peek(){
        if (isEmpty()) {
            System.out.println("Cannot peek from an empty 
stack!");
            return null;
        }

        return data[pointer];
}
```

a. yes

b. no

27. When a variable is defined within the body of a function is said to be in a__ scope.

    a. global

    b. local

    c. function

    d. internal

28. A data structure that stores items in a key-value pair in Java is said to be__

    a. Set

    b. Queue

    c. Map

    d. ArrayList

29. All the following are linear data structures except.

    a. Tree

    b. ArrayList

    c. Stack

    d. LinkedList

30. Iterating through a linear data structure, the time complexity of an algorithm depends on its size.

    a. true

    b. false

Section B:

Answer only two questions from this section. All questions carry equal marks.

1. The algorithms below create an array twice the size of the `data` array called `temp` and fill it with all the items from the `data` array. Suppose the `data` array consists of the following items; `[12, 4, 10, 19, 3]`.

```
int[] temp = new int[this.data.length * 2];
for (int i = 0; i < this.data.length; i++){
     temp[i] = data[i];
}
data = temp; // data is reassigned with temp
```

   a. Explain why the temp array will be twice the size of the data array. (2 marks)

   b. Explain how the items from the `data` array will copied to the `temp` array. (2 marks)

   c. Finally, output the content of the data array. (1 marks)

2. List all the three interfaces defined by the Collection Framework. (1 marks)

   a. Discuss the difference between any two. (2 marks)

   b. Use the one mentioned above and perform two operations(add and remove). (2 marks)

3. List the three iterative mechanisms discussed so far.

```
String[][] studentGrade = {
                           {"ahmed dicko", "4.0"},
                           {"binta jallow", "3.35"},
                           {"yuspha jarju", "2.75"},
                           {"jainaba njie", "3.75"},
                            };
```

   a. Explain why such a data structure(nested array) is the best suite for storing student grades. (2 marks)

   b. Java Collection provided a Data structure that suffices the need for handling such. Use the data structure to store the student grades. (2 marks)

   c. Which iterative mechanism will you use to traverse/iterate through the `studentGrades`? (1 marks)

4. You are tasked with storing information about students such as `id`, `name`,`age`, `gpa`, and `major` into a single type. Object-oriented programming allows us to create one single type(user-defined type) with many properties and can also define behaviors. A Student will have the above properties and a promote behavior.

   Consider this template:

```
public class Student {


}
```

   a. Using; `id, name, age, gpa, major`; Create field types. (1 marks)

   b. Create a constructor to initialize the instance variables. (1 marks)

c. As for behavior, create a function that will define how a student should be promoted. Hint: Students whose GPA is more than 1.9 are only eligible to be promoted. (1 marks)

d. An `ArrayList<Student> studentList` can be store students. After creating at least three students(object), make sure one student's `gpa` is less than 1.9, and add them to `studentList`. (1 marks)

e. Using a `for (Student in studentList)` to traverse the list, print only the student whose `gpa` is above 1.9. (1 marks)

5. A LinkedList data structure consists of at least a head, tail, and a Node class.

```
public class LL {
    Node head;
    Node tail;

    private class Node {}
}
```

a. What is head and tail? (1 mark)

b. Where do the head and tail both point to when one node is created?(1 mark)

c. For a Doubly LinkedList, what are the properties of a Node class? (1 mark)

d. Draw Doubly LinkedList and Consider the following as the values; `20, 45, 12, 66, 98`. (2 marks)