**Q1.What is Exception in Java?**

**Ans:** When an exception occurs, the program will stop executing the current code and will jump to a special section of code called a catch block. The catch block will handle the exception and will attempt to resume the normal flow of the program.

Syntax:-

```
try {
  // Code that may throw a checked exception
} catch (Exception e) {
  // Handle the checked exception

}
```

**Q2.What is Exception Handling?**

**Ans:-** When an exception occurs, the program will stop executing the current code and will jump to a special section of code called a catch block. The catch block will handle the exception and will attempt to resume the normal flow of the program.

```
try {
  // Code that may throw a checked exception
} catch (Exception e) {
  // Handle the checked exception

}
```

**Q3.What is the difference between Checked and Unchecked Exceptions and Error?**

**Ans:** Checked exceptions are exceptions that must be handled by the programmer. This means that if a checked exception is thrown, the program must have a `catch` block that handles the exception. If the program does not have a `catch` block for a checked exception, the program will not compile.

Unchecked exceptions are exceptions that do not need to be handled by the programmer. This means that if an unchecked exception is thrown, the program does not have to have a `catch` block that handles the exception.

**Q4.What are the difference between throw and throws in Java?**

**Ans:** In Java, the `throw` keyword is used to throw an exception. The `throws` keyword is used to declare that a method can throw an exception.

The `throw` keyword is used to throw an exception explicitly. This means that the programmer is explicitly telling the program that an exception has occurred. The `throw` keyword takes an exception object as its argument.

The `throws` keyword is used to declare that a method can throw an exception. This means that the method may throw an exception, but it does not have to. The `throws` keyword takes a list of exception types as its argument.

**Q5.What is multithreading in Java? mention its advantages**

**Ans:** Multithreading is a Java feature that allows you to run multiple tasks simultaneously. This means that you can have multiple threads of execution running at the same time.

Threads are lightweight processes that can share the same memory space. This makes them efficient and easy to use.

Advantage: Increased performance

Improved responsiveness

Improved scalability

**Q6.Write a program to create and call a custom exception**

**Ans:**
```
class MyException(Exception):

  def __init__(self, message):
    super().__init__(message)


def raise_my_exception():
  """Raises a custom exception."""
  raise MyException("This is a custom exception.")


try:
  raise_my_exception()
except MyException as e:

  print(e)
```

**Q7.How can you handle exceptions in Java?**

**Ans:** Using a `try-catch` block: A `try-catch` block is a block of code that can be used to handle exceptions. The `try` block contains the code that could potentially throw an exception. The `catch` block contains the code that will be executed if an exception is thrown

```
int x = 10;

int y = 0;

try {
  int z = x / y;
} catch (ArithmeticException e) {
  System.out.println("Division by zero.");

}
```

**Q8.What is Thread in Java?**

**Ans:** In Java, threads are created by extending the `Thread` class. The `Thread` class provides methods for starting, stopping, and suspending threads.

```
class MyThread extends Thread {
  public void run() {
    // Do something in the thread
  }
}

public class Main {
  public static void main(String[] args) {
    MyThread thread = new MyThread();
    thread.start();
  }
}
```

**Q10.What do you mean by garbage collection?**

**Ans:** Garbage collection (GC) is a automatic process in Java that deallocates memory that is no longer in use by an object. This is done by the Java Virtual Machine (JVM) and it is transparent to the programmer.

In Java, memory is allocated to objects on the heap. When an object is no longer in use, it is considered garbage and can be garbage collected. The JVM will periodically scan the heap for garbage objects and deallocate their memory.