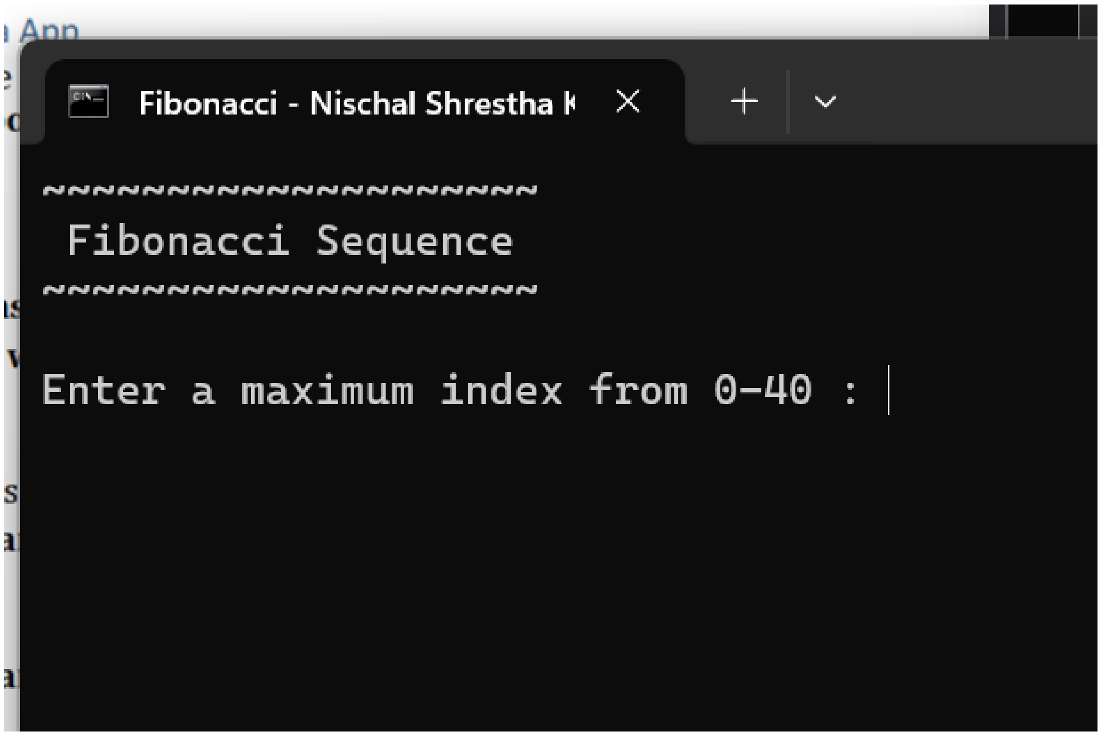
Object Oriented Programming

Prof. Fred Stiebler

ICE-5

Nischal Shrestha Kasula

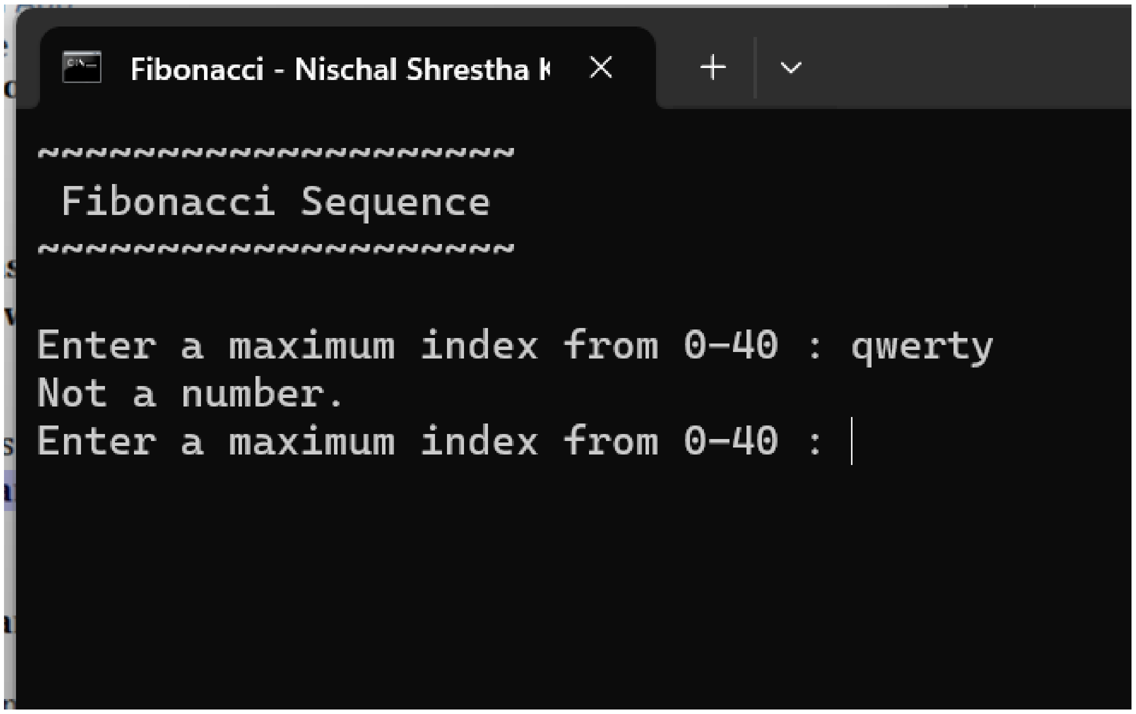
SS 1: Showing the initial screen, which is what you see as soon as the app is open.



```
~~~~~~~~~~~~~~~~~~~~~~
 Fibonacci Sequence
~~~~~~~~~~~~~~~~~~~~~~

Enter a maximum index from 0-40 :
```

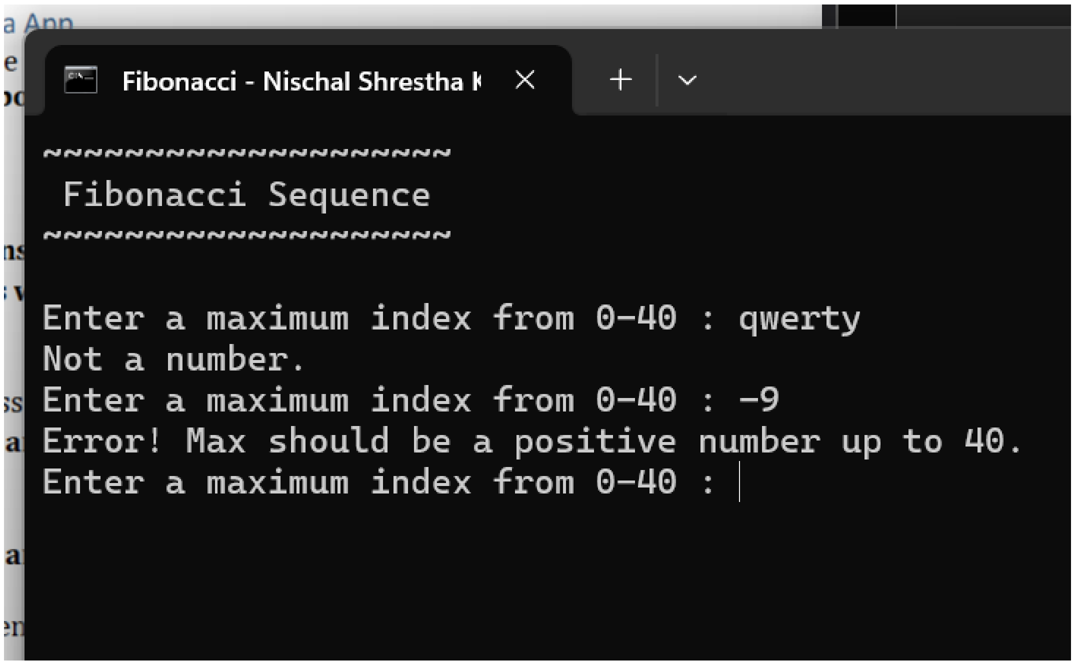
SS 2: Showing the error message when entered non numeric and the next prompt



```
~~~~~~~~~~~~~~~~~~~~~~
 Fibonacci Sequence
~~~~~~~~~~~~~~~~~~~~~~

Enter a maximum index from 0-40 : qwerty
Not a number.
Enter a maximum index from 0-40 :
```

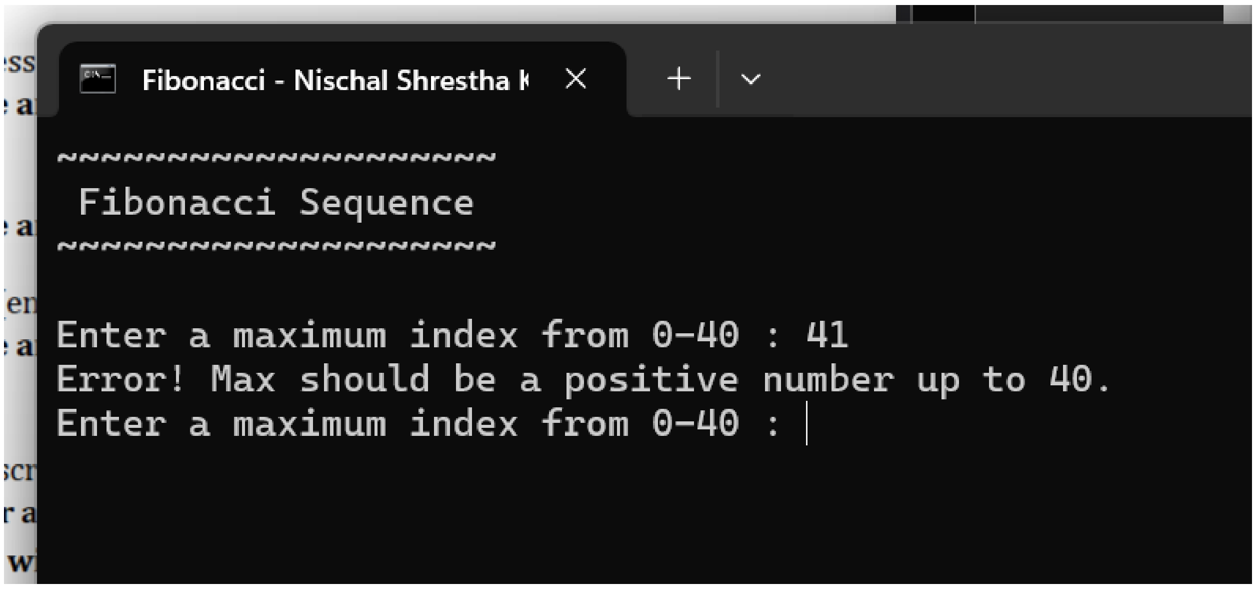
SS 3: Showing the error message when entered negative number and the next prompt



```
~~~~~~~~~~~~~~~~~~~~~
 Fibonacci Sequence
~~~~~~~~~~~~~~~~~~~~~

Enter a maximum index from 0-40 : qwerty
Not a number.
Enter a maximum index from 0-40 : -9
Error! Max should be a positive number up to 40.
Enter a maximum index from 0-40 :
```

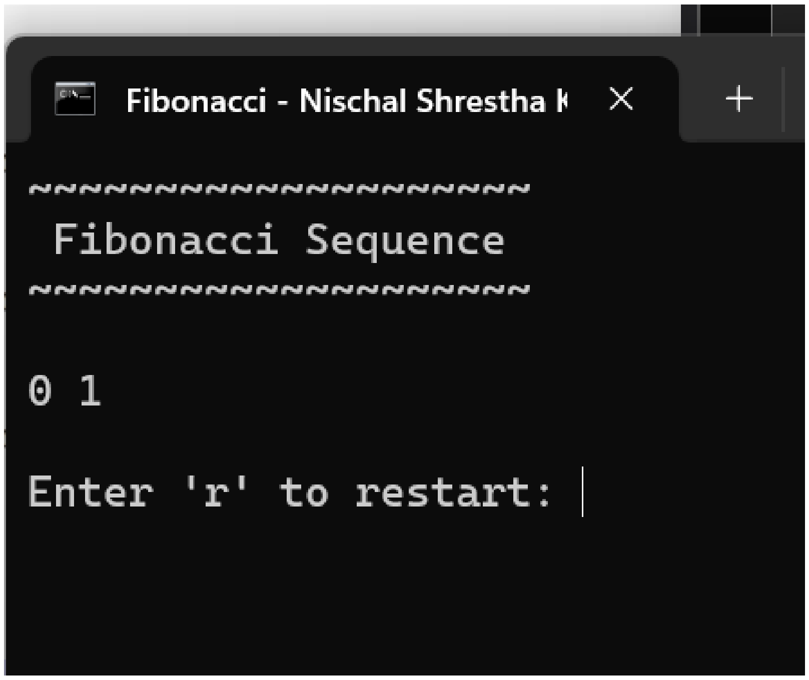
SS 4 : Showing the error message when entered number greater than 40 and the next prompt



```
~~~~~~~~~~~~~~~~~~~~~
 Fibonacci Sequence
~~~~~~~~~~~~~~~~~~~~~

Enter a maximum index from 0-40 : 41
Error! Max should be a positive number up to 40.
Enter a maximum index from 0-40 :
```

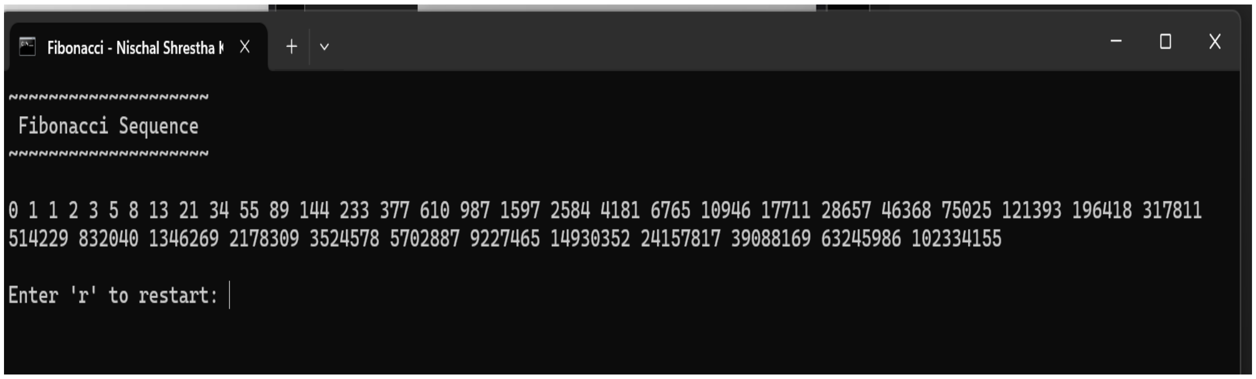
SS 5 & 6 : Showing the output screen with the exit prompt when entering valid inputs



```
~~~~~~~~~~~~~~~~~~~~
 Fibonacci Sequence
~~~~~~~~~~~~~~~~~~~~

0 1

Enter 'r' to restart:
```



```
~~~~~~~~~~~~~~~~~~~~
 Fibonacci Sequence
~~~~~~~~~~~~~~~~~~~~

0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 6765 10946 17711 28657 46368 75025 121393 196418 317811
514229 832040 1346269 2178309 3524578 5702887 9227465 14930352 24157817 39088169 63245986 102334155

Enter 'r' to restart:
```

❓ QUESTION 1 – What is a recursive method?
A function that calls itself to solve a problem, breaking it into smaller subproblems until reaching a base case.

❓ QUESTION 2 – Just like loops, what could go wrong if recursion is used incorrectly?
Like infinite loops, recursion without a base case leads to infinite recursion, causing memory overflow and crashes.

❓ QUESTION 3 – How can recursive methods avoid the problem in Question2?
Ensure a base case to stop recursion, reduce problem size with each call, and limit recursion depth if needed.