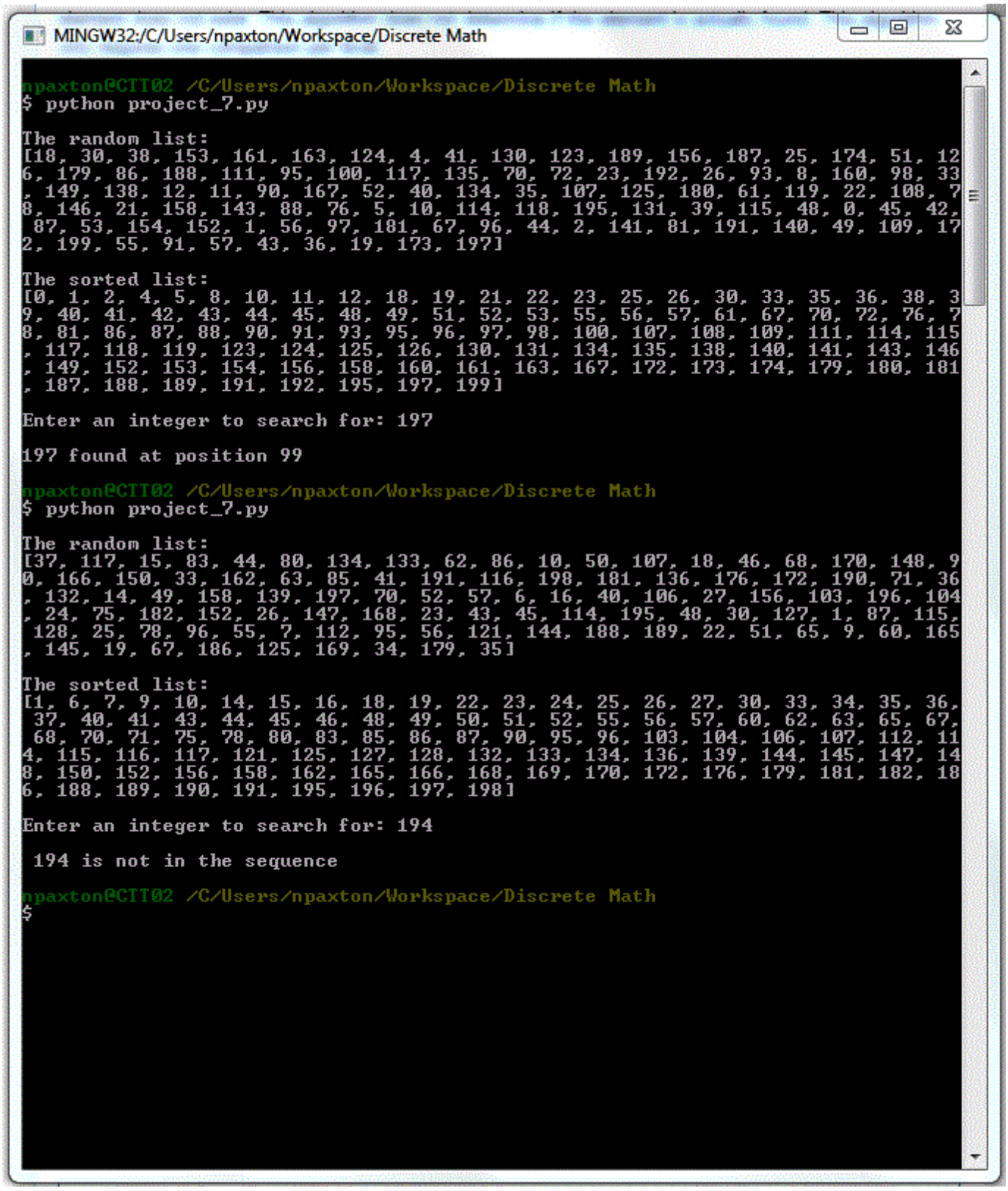
```
"""
The purpose of this program is to perform a binary search within a
randomly generated list of integers between 0 and 200.

The program will first generate a list of integers using the random
generator from project 5, and will sort the list using the insert
sort from project 6.  Binary searches can only be performed on sorted
lists.

The program will return the position of the integer being searched
for if it is present, and will return a message if not found.
"""
import random

class projectSeven:

    def __init__(self):

        #Generate the random number list
        ints = self.irand(100, 200)
        print("\nThe random list:\n" + str(ints))

        #Sort the list
        ints = self.insSort(ints)
        print("\nThe sorted list:\n" + str(ints))

        #Search for the index
        c = int(input("\nEnter an integer to search for: "))
        print(self.binSearch(ints, c))

    def binSearch(self, nums, c):

        #Variables to represent the start and end point, control logic
        a = 0
        b = len(nums) - 1
        found = False

        #Iterate over the values if not found
        while a <= b and not found:

            #Determine the midpoint for the search
            mid = int((a + b) // 2)

            #If the number matches, found == true
            if nums[mid] == c:
                found = True

            #Increment or decrement the upper and lower bounds depending
            #on comparison
            else:
                if c < nums[mid]:
                    b = mid - 1
                else:
                    a = mid + 1
```

```
55
56          if found:
57              return "\n" + str(c) + " found at position " + str(mid + 1)
58
59          else:
60              return "\n " + str(c) + " is not in the sequence"
61
62      def irand(self, n, m):
63          b = list(range(n))
64          b = random.sample(range(m), n)
65          return b
66
67      #Sort the list using an insertion sort
68      def insSort(self, nums):
69
70          #Compare the positions in the array
71          for i in range(1, len(nums)):
72
73              #The value to be compared
74              currentvalue = nums[i]
75
76              #Assign the iterator to a new variable to avoid index errors
77              position = i
78
79              #Position must be greater than zero so the index can't be -1
80              while position > 0 and nums[position - 1] > currentvalue:
81
82                  #Assign the value to a new position
83                  nums[position] = nums[position - 1]
84                  position = position - 1
85
86              #Assign the value to a new position
87              nums[position] = currentvalue
88
89          #Return the sorted array
90          return nums
91
92  p = projectSeven()
```


MINGW32:/C/Users/npaxton/Workspace/Discrete Math
npaxton@CTT02 /C/Users/npaxton/Workspace/Discrete Math
$ python project_7.py
The random list:
[18, 30, 38, 153, 161, 163, 124, 4, 41, 130, 123, 189, 156, 187, 25, 174, 51, 12
6, 179, 86, 188, 111, 95, 100, 117, 135, 70, 72, 23, 192, 26, 93, 8, 160, 98, 33
, 149, 138, 12, 11, 90, 167, 52, 40, 134, 35, 107, 125, 180, 61, 119, 22, 108, 7
8, 146, 21, 158, 143, 88, 76, 5, 10, 114, 118, 195, 131, 39, 115, 48, 0, 45, 42,
87, 53, 154, 152, 1, 56, 97, 181, 67, 96, 44, 2, 141, 81, 191, 140, 49, 109, 17
2, 199, 55, 91, 57, 43, 36, 19, 173, 197]
The sorted list:
[0, 1, 2, 4, 5, 8, 10, 11, 12, 18, 19, 21, 22, 23, 25, 26, 30, 33, 35, 36, 38, 3
9, 40, 41, 42, 43, 44, 45, 48, 49, 51, 52, 53, 55, 56, 57, 61, 67, 70, 72, 76, 7
8, 81, 86, 87, 88, 90, 91, 93, 95, 96, 97, 98, 100, 107, 108, 109, 111, 114, 115
, 117, 118, 119, 123, 124, 125, 126, 130, 131, 134, 135, 138, 140, 141, 143, 146
, 149, 152, 153, 154, 156, 158, 160, 161, 163, 167, 172, 173, 174, 179, 180, 181
, 187, 188, 189, 191, 192, 195, 197, 199]
Enter an integer to search for: 197
197 found at position 99
npaxton@CTT02 /C/Users/npaxton/Workspace/Discrete Math
$ python project_7.py
The random list:
[37, 117, 15, 83, 44, 80, 134, 133, 62, 86, 10, 50, 107, 18, 46, 68, 170, 148, 9
0, 166, 150, 33, 162, 63, 85, 41, 191, 116, 198, 181, 136, 176, 172, 190, 71, 36
, 132, 14, 49, 158, 139, 197, 70, 52, 57, 6, 16, 40, 106, 27, 156, 103, 196, 104
, 24, 75, 182, 152, 26, 147, 168, 23, 43, 45, 114, 195, 48, 30, 127, 1, 87, 115,
128, 25, 78, 96, 55, 7, 112, 95, 56, 121, 144, 188, 189, 22, 51, 65, 9, 60, 165
, 145, 19, 67, 186, 125, 169, 34, 179, 35]
The sorted list:
[1, 6, 7, 9, 10, 14, 15, 16, 18, 19, 22, 23, 24, 25, 26, 27, 30, 33, 34, 35, 36,
37, 40, 41, 43, 44, 45, 46, 48, 49, 50, 51, 52, 55, 56, 57, 60, 62, 63, 65, 67,
68, 70, 71, 75, 78, 80, 83, 85, 86, 87, 90, 95, 96, 103, 104, 106, 107, 112, 11
4, 115, 116, 117, 121, 125, 127, 128, 132, 133, 134, 136, 139, 144, 145, 147, 14
8, 150, 152, 156, 158, 162, 165, 166, 168, 169, 170, 172, 176, 179, 181, 182, 18
6, 188, 189, 190, 191, 195, 196, 197, 198]
Enter an integer to search for: 194
194 is not in the sequence
npaxton@CTT02 /C/Users/npaxton/Workspace/Discrete Math
$