System Research Skills Assessment

Q1. Please find the outcome of print and deep clone function below.

void print(k) function example:

Q1.cpp Cosmos interview questions.docx.pdf

Q1.cpp > main()

```cpp
class FirstNFibonacci {
    ~FirstNFibonacci() {
        while (curr) {
            LinkedListNode* nextNode = curr->next;
            delete curr;
            curr = nextNode;
        }
    }
};

int main() {
    FirstNFibonacci myObj(6);  // Create original linked list
    myObj.print(99);
    return 0;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
tejassinghal@Tejass-MacBook-Pro-2 Tejas Singhal Research work with Dr. Khanh Nguyen % ./a.out
The first 99 nodes of the linked list are:
0
1
1
2
3
5
tejassinghal@Tejass-MacBook-Pro-2 Tejas Singhal Research work with Dr. Khanh Nguyen %
```

To verify that the cloned linked list is not the original, you need to ensure that:

- The values are the same
- The memory addresses are different

Q1.cpp Cosmos interview questions.docx.pdf

Q1.cpp > ...

```cpp
int main() {
    FirstNFibonacci myObj(6);  // Create original linked list
    FirstNFibonacci* clonedObj = myObj.deep_clone();  // Create cloned linked list

    LinkedListNode* originalNode = myObj.head;
    LinkedListNode* clonedNode = clonedObj->head;

    cout << "Verifying deep clone by comparing node addresses:\n";
    while (originalNode && clonedNode) {
        cout << "Original Node Address: " << originalNode << " | Value: " << originalNode->val << endl;
        cout << "Cloned Node Address:   " << clonedNode << " | Value: " << clonedNode->val << endl;

        cout << endl;

        originalNode = originalNode->next;
        clonedNode = clonedNode->next;
    }

    delete clonedObj;
    return 0;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
tejassinghal@Tejass-MacBook-Pro-2 Tejas Singhal Research work with Dr. Khanh Nguyen % g++ Q1.cpp
tejassinghal@Tejass-MacBook-Pro-2 Tejas Singhal Research work with Dr. Khanh Nguyen % ./a.out
Verifying deep clone by comparing node addresses:
Original Node Address: 0x121605f60 | Value: 0
Cloned Node Address:   0x121606070 | Value: 0

Original Node Address: 0x121605f70 | Value: 1
Cloned Node Address:   0x121605fc0 | Value: 1

Original Node Address: 0x121605ee0 | Value: 1
Cloned Node Address:   0x121605fd0 | Value: 1

Original Node Address: 0x121605ef0 | Value: 2
Cloned Node Address:   0x121605fe0 | Value: 2

Original Node Address: 0x121606040 | Value: 3
Cloned Node Address:   0x121605ff0 | Value: 3
```

```
Original Node Address: 0x121606050 | Value: 5
Cloned Node Address:   0x121605cc0 | Value: 5
```

As it can be seen above, the original Node Address and Cloned Node Address are different but the values are the same, indicating deep cloning happened.

Q2. Algorithm**:**

1. Compare elements in pairs instead of individually.
2. Track the larger element as a candidate for max and the smaller element as a candidate for min.
3. Only one comparison per two elements to decide which one is larger, and which one is smaller.
4. After deciding their roles, compare them with the current max and min.

Number of Comparisons:

- We group elements into pairs and do one comparison per pair → (n/2) comparisons.
- Each of the two winners (largest and smallest in each pair) is compared with the current max or min → (n - 2) comparisons.
- Total comparisons: n2+(n−2) = (3n/2) – 2 = 1.5n – 2 < 2n – 2

The above is the tournament method to find the max and min in a given list of integers.