

```
void printPowerset(int n, int *bestSet, int &bestSize, float *A) {
```

```
    int *stack, k; 2
```

$$Rt. \boxed{7 + 2^n 5k + 2^n 7}$$

```
    // allocate space for the set
```

```
    stack = new int[n + 1]; 1
```

```
    stack[0] = 0; 1
```

```
    k = 0;
```

```
    while (1) {
```

```
        if (stack[k] < n) {
```

```
            stack[k + 1] = stack[k] + 1; 3
```

```
            k++;
```

```
        } else {
```

```
            stack[k - 1]++; 3
```

```
            k--;
```

```
        }
```

```
        if (k == 0)
```

```
            break; 1
```

```
        checkSet(stack, k, bestSet, bestSize, A); 5k+3
```

```
    delete[] stack; 1
```

```
    return; 1
```

```
void checkSet(int *stack, int k, int *bestSet, int &bestSize, float *A) { checkSet Rt = 5k+3
// function to check the currently generated set stack of size k against the current
// best set bestSet of size bestSize
```

```
    int i; 1
```

```
    if (k < 2) {
```

```
        if (k > bestSize) {
```

```
            for (i = 0; i <= k; i++)
```

```
                bestSet[i] = stack[i];  $\sum_{i=0}^k 1 \cdot (k-0+1) = k+4$ 
```

```
            bestSize = k;
```

```
            return;
```

```
        } else {
```

```
            for (i = 0; i < k - 1; i++) {
```

```
                if (A[stack[i + 1] - 1] > A[stack[i + 2] - 1]) 3 +  $\sum_{i=0}^{k-2} 4 = 3 + 4(k-2-0+1) = 4k-4$ 
```

```
                    return;
```

```
            }
```

```
        if (k > bestSize) {
```

```
            for (i = 0; i <= k; i++)
```

```
                bestSet[i] = stack[i];  $3 + \sum_{i=0}^k 1 = 3 + 1(k-0+1) = k+4$ 
```

```
            bestSize = k;
```

```
        return;
```

$$1 + 4k - 3 + k + 5 = 5k + 3$$

$$\text{PowerSet Rt } 2^n 5k + 2^n 7 + 7 \in O(2^n k)$$

$$2^n 5k + 2^n 7 + 7 \leq C 2^n k \quad \forall n \geq n_0$$

$$\text{Let } C = 100 \quad 5 \cdot 2^n k + 2^n 7 + 7 \leq 100 \cdot 2^n k \quad \forall n \geq n_0$$

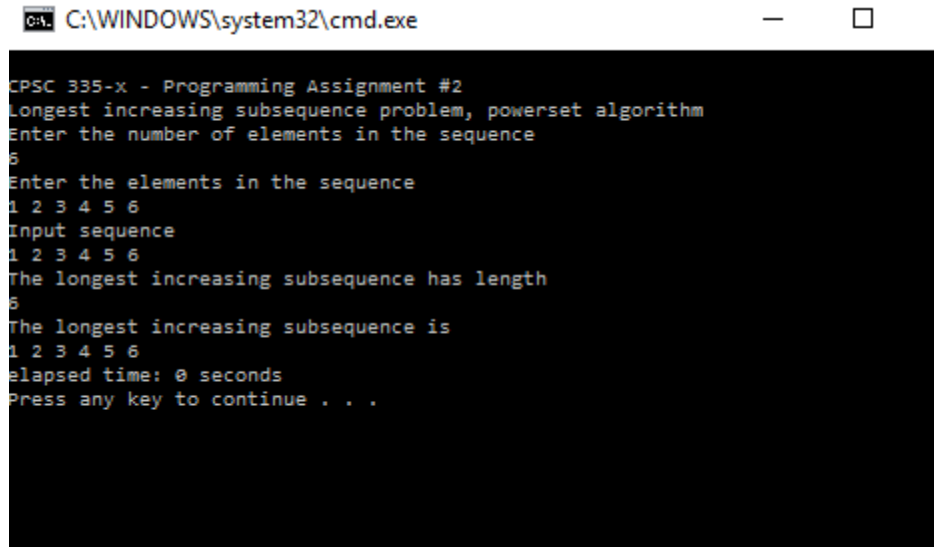
$$7 \cdot 2^n + 7 \leq 95 \cdot 2^n k \quad \forall n \geq n_0$$

$$\text{Let } n_0 = 1$$

$$21 \leq 190 k \quad \forall n \geq 1$$

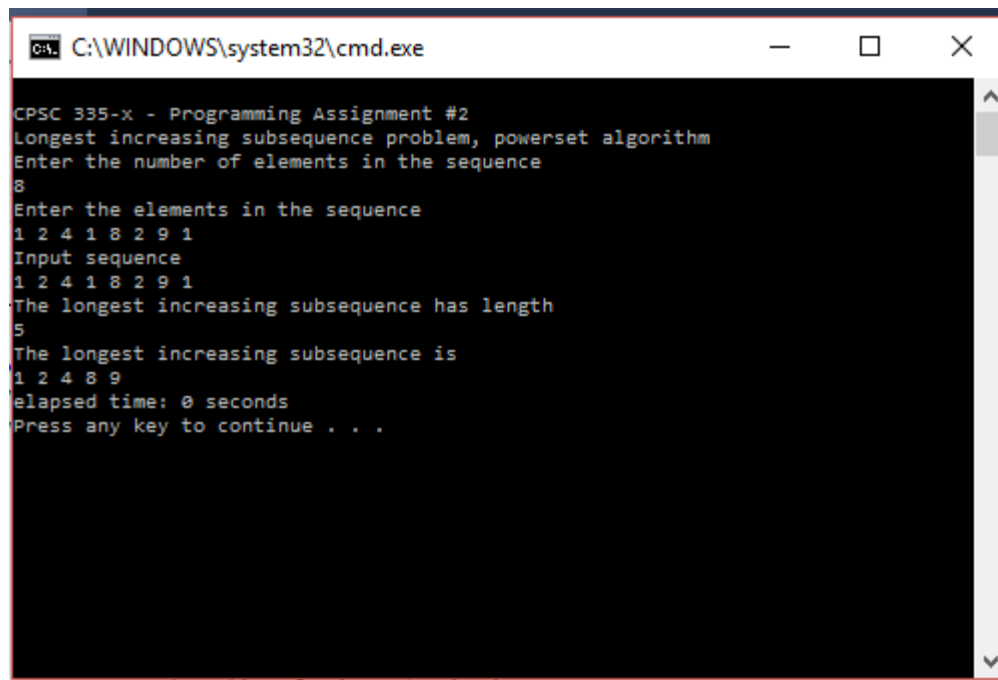
therefore
PowerSet $O(2^n k)$

Outputs :



```
C:\WINDOWS\system32\cmd.exe

CPSC 335-x - Programming Assignment #2
Longest increasing subsequence problem, powerset algorithm
Enter the number of elements in the sequence
6
Enter the elements in the sequence
1 2 3 4 5 6
Input sequence
1 2 3 4 5 6
The longest increasing subsequence has length
6
The longest increasing subsequence is
1 2 3 4 5 6
elapsed time: 0 seconds
Press any key to continue . . .
```



```
C:\WINDOWS\system32\cmd.exe

CPSC 335-x - Programming Assignment #2
Longest increasing subsequence problem, powerset algorithm
Enter the number of elements in the sequence
8
Enter the elements in the sequence
1 2 4 1 8 2 9 1
Input sequence
1 2 4 1 8 2 9 1
The longest increasing subsequence has length
5
The longest increasing subsequence is
1 2 4 8 9
elapsed time: 0 seconds
Press any key to continue . . .
```