

# VARUN BHARGAVA – 241010282

## DATA STRUCTURES TASK-8

### Task 01: Josephus Problem:

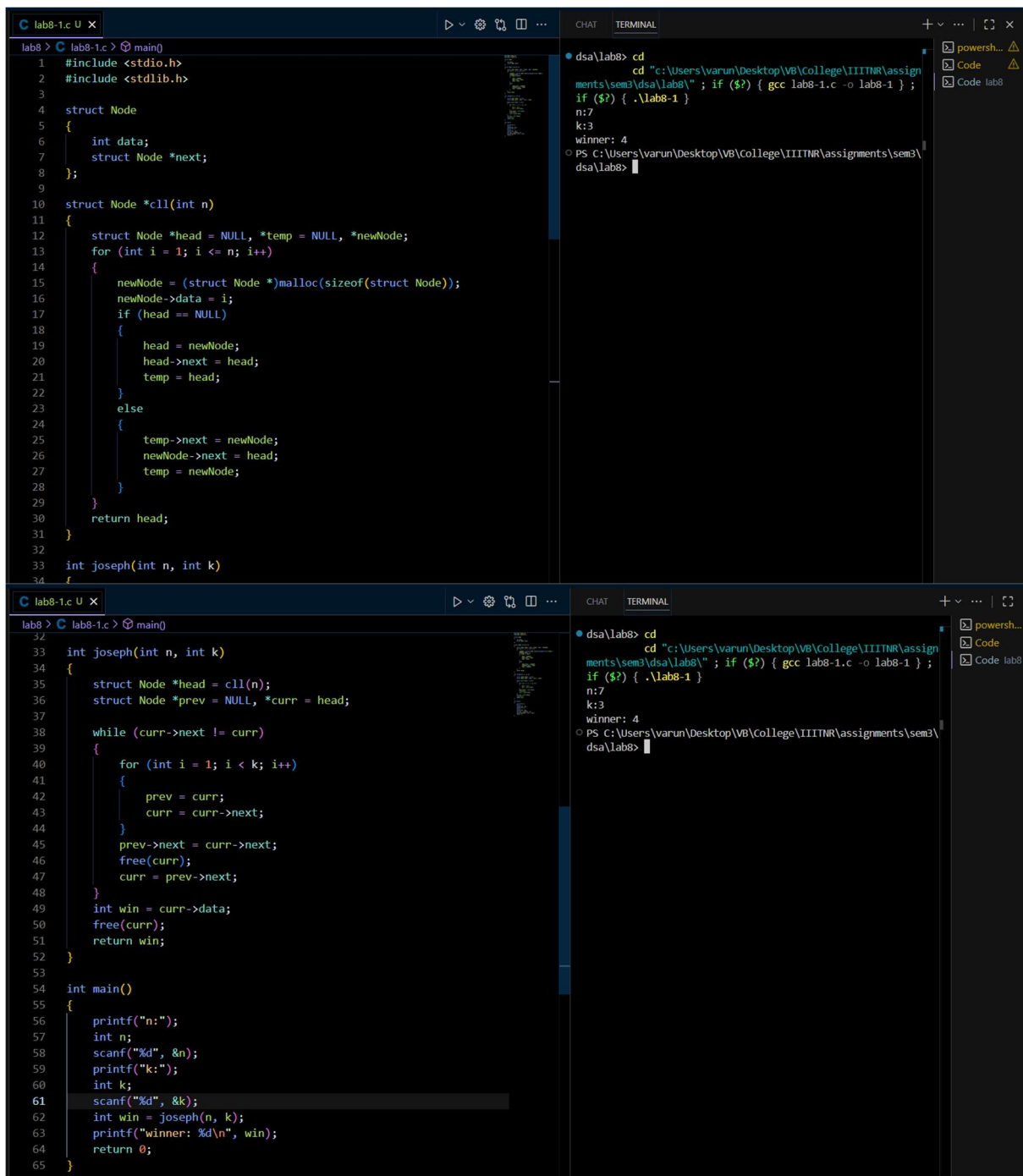
(<https://github.com/varunnnb/dsa-sem3-iiitnr/blob/main/lab8/lab8-1.c>)

Write a program that simulates the Josephus Problem, where n players stand in a circle and every kth player are eliminated until only one player remains. The program should determine and print the position of the winner (survivor).

Example:

Input: n = 7, k = 3

Output: The winner is player 4



```
lab8-1.c U X
lab8 > C lab8-1.c > main()
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 struct Node
5 {
6     int data;
7     struct Node *next;
8 };
9
10 struct Node *c1l(int n)
11 {
12     struct Node *head = NULL, *temp = NULL, *newNode;
13     for (int i = 1; i <= n; i++)
14     {
15         newNode = (struct Node *)malloc(sizeof(struct Node));
16         newNode->data = i;
17         if (head == NULL)
18         {
19             head = newNode;
20             head->next = head;
21             temp = head;
22         }
23         else
24         {
25             temp->next = newNode;
26             newNode->next = head;
27             temp = newNode;
28         }
29     }
30     return head;
31 }
32
33 int joseph(int n, int k)
34 {
35     struct Node *head = c1l(n);
36     struct Node *prev = NULL, *curr = head;
37
38     while (curr->next != curr)
39     {
40         for (int i = 1; i < k; i++)
41         {
42             prev = curr;
43             curr = curr->next;
44         }
45         prev->next = curr->next;
46         free(curr);
47         curr = prev->next;
48     }
49     int win = curr->data;
50     free(curr);
51     return win;
52 }
53
54 int main()
55 {
56     printf("n:");
57     int n;
58     scanf("%d", &n);
59     printf("k:");
60     int k;
61     scanf("%d", &k);
62     int win = joseph(n, k);
63     printf("winner: %d\n", win);
64     return 0;
65 }
```

```
dsa\lab8> cd
cd "c:\Users\varun\Desktop\VB\College\IIITNR\assignments\sem3\dsa\lab8\" ; if ($?) { gcc lab8-1.c -o lab8-1 } ;
if ($?) { .\lab8-1 }
n:7
k:3
winner: 4
PS C:\Users\varun\Desktop\VB\College\IIITNR\assignments\sem3\dsa\lab8>
```

## Task 02: Single Linked List – Longest Ascending Sequence:

( <https://github.com/varunnrb/dsa-sem3-iiitnr/blob/main/lab8/lab8-2.c> )

Write a program using a single linked list that takes input from the user and prints the longest sequence of ascending order integers in it along with the length.

Example:

Input: 20, 2, 5, 17, 77, 7, 5, 3

Output: Longest ascending sequence: 2 5 17 77

```
lab8-2.c> createNode(int)
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 struct Node
5 {
6     int data;
7     struct Node *next;
8 };
9
10 struct Node *createNode(int data)
11 {
12     struct Node *newNode = (struct Node *)malloc(sizeof(struct Node));
13     newNode->data = data;
14     newNode->next = NULL;
15     return newNode;
16 }
17
18 void append(struct Node **head, int data)
19 {
20     struct Node *newNode = createNode(data);
21     if (*head == NULL)
22     {
23         *head = newNode;
24         return;
25     }
26     struct Node *temp = *head;
27     while (temp->next != NULL)
28     {
29         temp = temp->next;
30     }
31     temp->next = newNode;
32 }
33
34 void longest(struct Node *head)
35 {
36     if (head == NULL)
37     {
38         printf("list empty.\n");
39         return;
40     }
41
42     struct Node *curr = head;
43     struct Node *start = head;
44     struct Node *bestStart = head;
45
46     int length = 1, bestLength = 1;
47
48     while (curr->next != NULL)
49     {
50         if (curr->next->data > curr->data)
51         {
52             length++;
53         }
54         else
55         {
56             if (length > bestLength)
57             {
58                 bestLength = length;
59                 bestStart = start;
60             }
61             start = curr->next;
62             length = 1;
63         }
64         curr = curr->next;
65     }
66
67     if (length > bestLength)
68     {
69         bestLength = length;
70         bestStart = start;
71     }
72
73     printf("longest seq: ");
74     curr = bestStart;
75     for (int i = 0; i < bestLength; i++)
76     {
77         printf("%d ", curr->data);
78         curr = curr->next;
79     }
80 }
```

```
PS C:\Users\varun\Desktop\VB\College\IIITNR\assignments\sem3\dsa\lab8> cd "c:\Users\varun\Desktop\VB\College\IIITNR\assignments\sem3\dsa\lab8"; if ($?) { gcc lab8-2.c -o lab8-2 }
if ($?) { .\lab8-2 }
Enter number of elements: 6
Enter 6 integers:
12
14
31
43
12
17
longest seq: 12 14 31 43
Length = 4
PS C:\Users\varun\Desktop\VB\College\IIITNR\assignments\sem3\dsa\lab8>
```

```
lab8-1.c U lab8-2.c U X lab8-3.c U
lab8 > C lab8-2.c > createNode(int)
34 void longest(struct Node *head)
75     for (int i = 0; i < bestLength; i++)
76     {
77         printf("%d ", curr->data);
78         curr = curr->next;
79     }
80     printf("\nLength = %d\n", bestLength);
81 }
82
83 int main()
84 {
85     struct Node *head = NULL;
86     int n, value;
87
88     printf("Enter number of elements: ");
89     scanf("%d", &n);
90
91     printf("Enter %d integers:\n", n);
92     for (int i = 0; i < n; i++)
93     {
94         scanf("%d", &value);
95         append(&head, value);
96     }
97
98     longest(head);
99
100     return 0;
101 }
102
```

```
PS C:\Users\varun\Desktop\VB\College\IIITNR\assignments\sem3\dsa\lab8> cd "c:\Users\varun\Desktop\VB\College\IIITNR\assignments\sem3\dsa\lab8" ; if ($?) { gcc lab8-2.c -o lab8-2 }
if ($?) { .\lab8-2 }
Enter number of elements: 6
Enter 6 integers:
12
14
31
43
12
17
longest seq: 12 14 31 43
Length = 4
PS C:\Users\varun\Desktop\VB\College\IIITNR\assignments\sem3\dsa\lab8>
```

## Task 03: Anagram Check:

( <https://github.com/varunnnb/dsa-sem3-iiitnr/blob/main/lab8/lab8-3.c> )

Check whether two input strings are anagrams or not

Example:

Input: listen, silent

Output: Strings are Anagrams

```
lab8-1.c U lab8-2.c U lab8-3.c U X
C lab8-3.c > ...
1 #include <stdio.h>
2 #include <string.h>
3
4 int areAnagrams(char str1[], char str2[])
5 {
6     int count[256] = {0};
7
8     if (strlen(str1) != strlen(str2))
9         return 0;
10
11     for (int i = 0; str1[i] && str2[i]; i++)
12     {
13         count[(unsigned char)str1[i]]++;
14         count[(unsigned char)str2[i]]--;
15     }
16
17     for (int i = 0; i < 256; i++)
18     {
19         if (count[i] != 0)
20             return 0;
21     }
22     return 1;
23 }
24
25 int main()
26 {
27     char str1[100], str2[100];
28
29     printf("Enter first string: ");
30     scanf("%s", str1);
31     printf("Enter second string: ");
32     scanf("%s", str2);
33
34     if (areAnagrams(str1, str2))
35         printf("Strings are Anagrams\n");
36     else
37         printf("Strings are Not Anagrams\n");
38
39     return 0;
40 }
41
```

```
dsa> cd "c:\Users\varun\Desktop\VB\College\IIITNR\assignments\sem3\dsa" ; if ($?) { gcc lab8-3.c -o lab8-3 } ; if ($?) { .\lab8-3 }
Enter first string: varun
Enter second string: runva
Strings are Anagrams
PS C:\Users\varun\Desktop\VB\College\IIITNR\assignments\sem3\dsa>
```