# C AND DATA STRUCTURE-ASSIGNMENT II

**NO:02** **SRIGIRI T**

**DATE:30/06/2021** **20EUCS137**

**QUESTION 1:**

Given a list, split it into two sublists — one for the front half, and one for the back half. If the number of elements is odd, the extra element should go in the front list. So FrontBackSplit() on the list {2, 3, 5, 7, 11} should yield the two lists {2, 3, 5} and {7,11}.

**CODE:**

```
#include<stdio.h>
void FrontBackSplit(){
    int arr[]={2, 3, 5, 7, 11},f_arr[50],s_arr[50],i,j,k;
    int N = sizeof(arr) / sizeof(int);
    printf("Given array\n{");
    for(k=0;k<N-1;k++)
        printf("%d,",arr[k]);
    printf("%d}\n",arr[k]);
    if(N%2==0){
        for(i=0;i<N/2;i++)
            f_arr[i]=arr[i];
        for(j=0;j<N;j++)
            s_arr[j]=arr[j+N/2];
    }
    else{
        for(i=0;i<N/2+1;i++)
            f_arr[i]=arr[i];
        for(j=0;j<N;j++)
            s_arr[j]=arr[j+N/2+1];
    }
    if(N%2==0){
        printf("First array\n{");
        for(i=0;i<N/2-1;i++)
            printf("%d, ", f_arr[i]);
```

```
        printf("%d",f_arr[i]);
        printf("}\nSecond array\n{");
        for(j=0;j<N/2-1;j++)
            printf("%d, ", s_arr[j]);
        printf("%d}",s_arr[j]);
    }
    else{
        printf("First array\n{");
        for(i=0;i<N/2;i++)
            printf("%d, ", f_arr[i]);
        printf("%d",f_arr[i]);
        printf("}\nSecond array\n{");
        for(j=0;j<N/2-1;j++)
            printf("%d, ", s_arr[j]);
        printf("%d}",s_arr[j]);
    }
}
int main(){
    FrontBackSplit();
    return 0;
}
```

**OUTPUT:**

```
Given array
{2,3,5,7,11}
First array
{2, 3, 5}
Second array
{7, 11}
```

**QUESTION 2:**

Insert a node in a sorted linked list
Example:

10 20 30 40 50

Element to insert: 35
**Output:** 10 20 30 35 40 50

**CODE:**

```
#include<stdio.h>
#include<stdlib.h>
struct node{
    int data;
    struct node *next;
};
int main(){
    int N=0,element,arr[]={10,20,30,40,50};
    struct node *head=NULL, *neww, *temp, *prev;
    do{
         neww = (struct node*)malloc(sizeof(struct node));
            neww -> data = arr[N];
            neww -> next = NULL;
            if(head == NULL){
                head = neww;
                temp = neww;
            }
            else{
                temp -> next = neww;
                temp = neww;
            }
            N++;
    }while(N<5);
    temp = head;
    printf("Linked list before insertion \n");
    while (temp!=NULL){
        printf("%d->", temp-> data);
```

```
        temp = temp ->next;
    }
    printf("NULL\n");
    printf("Enter the value to insert ");
    scanf("%d",&N);
    neww = (struct node*)malloc(sizeof(struct node));
    neww ->data = N;
    temp = head;
    while(temp !=NULL){
        if(temp->next->data > N)
            break;
        temp=temp->next;
    }
        neww->next = temp->next;
        temp->next = neww;
    temp = head;
    printf("Linked list after insertion \n");
    while (temp!=NULL){
        printf("%d->", temp-> data);
        temp = temp ->next;
    }
    printf("NULL");
}
```

**OUTPUT:**

```
Linked list before insertion
10->20->30->40->50->NULL
Enter the value to insert 35
Linked list after insertion
10->20->30->35->40->50->NULL
```

**QUESTION 3:**

Reverse a doubly linked list

Example:

10 20 30 40

**Output:** 40 30 20 10

**CODE:**

```
#include<stdio.h>
#include<stdlib.h>
struct node{
    int data;
    struct node *next,*prev;
};
int main(){
    int N=0,i=0,arr[]={10,20,30,40};
    struct node *head=NULL,*neww,*temp,*temp2,*last;
            do{
                neww = (struct node*)malloc(sizeof(struct node));
                neww -> data = arr[N];
                neww -> next = NULL;
                neww -> prev = NULL;
                if(head == NULL){
                    head = neww;
                    temp = head;
                }
                else{
                    temp -> next = neww;
                    neww -> prev = temp;
                    temp = neww;
                }
                N++;
        }while(N<4);
    while(temp->next!=NULL)
        temp=temp->next;
    last=temp;
```

```
    temp = head;
    printf("The doubly linked list \n");
    printf("NULL<=>");
    while (temp != NULL){
                printf("%d<=>", temp-> data);
                temp = temp ->next;
            }
            printf("NULL\n");
    temp=head;
    while(temp2 != NULL)
    {
        temp2 = temp->next;
        temp->next = temp->prev;
        temp->prev = temp2;
        temp = temp2;
    }
    temp = head;
    head = last;
    last = temp;
    temp=head;
    printf("Reversed Doubly linked list\nNULL<=>");
    while (temp != NULL){
                printf("%d<=>", temp-> data);
                temp = temp ->next;
            }
            printf("NULL");
}
```

**OUTPUT:**

```
The doubly linked list
NULL<=>10<=>20<=>30<=>40<=>NULL
Reversed Doubly linked list
NULL<=>40<=>30<=>20<=>10<=>NULL
```

**QUESTION 4:**

Display the circular linked list from the last

Example:

10 20 30

**Output:** 30 20 10

**CODE:**

```
#include<stdio.h>
#include<stdlib.h>
struct node{
    int data;
    struct node *next;
};
int count=0;
int main(){
    int N=0,arr[]={10,20,30,40,50},i;
    struct node *head=NULL,*temp,*neww,*prev=NULL,*temp2;
    do{
        neww = (struct node*)malloc(sizeof(struct node));
        neww -> data = arr[N];
            if(head == NULL){
                head = neww;
                neww -> next = head;
                temp = head;
            }
            else{
                temp -> next = neww;
                neww -> next = head;
                temp = neww;
            }
        count++;
        N++;
    }while(N<5);
    temp = head;
            printf("->");
```

```
            while (temp->next != head)
            {
                printf("%d->", temp-> data);
                temp = temp ->next;
            }
            printf("%d-", temp->data);
            printf("\n|");
            for(i=0;i<N*4;i++)
                printf("_");
            printf("| Circular Linked List\n");
    printf("->");
    ReverseCircle(head);
    printf("\n|");
    for(int k=0;k<N*4;k++)
        printf("_");
    printf("| Circular Linked List form the last ");
}
void ReverseCircle(struct node* head){
    if (count--==0)
        return;
    ReverseCircle(head->next);
    printf("%d->", head->data);
}
```

**OUTPUT:**

```
->10->20->30->40->50-
|____________________| Circular Linked List
->50->40->30->20->10->
|____________________| Circular Linked List form the last
```

**QUESTION 5:**

Validate the given symbols using stack

Example:

{{()(){{}}}}

**Output: Valid**

**CODE:**

```
#include<stdio.h>
#include<string.h>
int main(){
    char stack[100];
    printf("Enter the string : ");
    scanf("%s", stack);
    int len = strlen(stack);
    int i=0,top=-1;
    while(i<len){
        if(stack[i]=='{'||stack[i]=='('||stack[i]=='[')
            stack[++top] = '{';
        else if(stack[i]=='}'||stack[i]==')'||stack[i]==']'){
            if(top==-1){
                printf("Invalid");
                return 0;
            }
            else
                top--;
        }
    i++;
    }
    if(top==-1)
        printf("Valid");
}
```

**OUTPUT:**

```
Enter the string : {{()[]{{}}}}
Valid
```