

PROGRAM 11

```
/* **** */
/* Name of the Program      : josephus.c          */
/* Aim                     : To solve josephus problem */
/* Author                  : ASWIN BABU          */
/* Date Written            : 23/03/2017         */
/* Revision                 : 1                  */
/* **** */
```

```
/* **** */
/* PROGRAM:                                     */
/* survive                 : To store the survivor */
/* skip                    : No:of persons to be skipped */
/* head                    : Head part of the node  */
/* **** */
```

```
#include<stdio.h>
#include<stdlib.h>
struct node
{
    int num;
    struct node *next;
};
void create(struct node**);
void display(struct node*);
int survivor(struct node**,int);

int main()
{
    struct node *head=NULL;
    int survive,skip;
    create(&head);
    printf("The people in the circular queue are:\n");
    display(head);
    printf("Enter the number of persons to be skipped\n");
    scanf("%d",&skip);
    survive=survivor(&head,skip);
    printf("The person to survive is %d\n",survive);
    free(head);
}
int survivor(struct node **head,int k)
{
    struct node *p,*q;
    int i;
    q=p=*head;
    while(p->next!=p)
```

```

    {
        for(i=0;i<k-1;i++)
        {
            q=p;
            p=p->next;
        }
        q->next=p->next;
        printf("%d has been killed.\n",p->num);
        free(p);
        p=q->next;
    }
    *head=p;
    return(p->num);
}

```

void create(struct node **head)

```

{
    struct node *temp,*rear;
    int a,ch;
    do
    {
        printf("Enter a number:");
        scanf("%d",&a);
        temp=(struct node *)malloc(sizeof(struct node));
        temp->num=a;
        temp->next=NULL;
        if(*head==NULL)
            *head=temp;
        else
            rear->next=temp;
        rear=temp;
        printf("Do you want to add a number? Press 1 to continue and 0 to stop");
        scanf("%d",&ch);
    }while(ch!=0);
    rear->next=*head;
}

```

void display(struct node *head)

```

{
    struct node *temp;
    temp=head;
    printf("%d ",temp->num);
    temp=temp->next;
    while(head!=temp)
    {

```

```
        printf("%d ",temp->num);
        temp=temp->next;
    }
    printf("\n");
}
}
}
```

```
/***/
```

```
/*  
*/
```

OUTPUT:

```
Enter a number: 3  
Do you want to add :1  
Enter a number: 6  
Do you want to add :1  
Enter a number: 2  
Do you want to add :1  
Enter a number: 7  
Do you want to add :1  
Enter a number: 5  
Do you want to add :1  
Enter a number: 1  
Do you want to add :1  
Enter a number: 4  
Do you want to add :0  
The People in the queue: 3 6 2 7 5 1 4  
Enter the number of persons to be skipped : 3  
2 has killed  
1 has killed  
6 has killed  
4 has killed  
5 has killed  
3 has killed  
The person to survive : 7  
*/  
/*
```

PROGRAM 12

```

/*****
/* Name of the Program      : poly.c                      */
/* Aim                     : To add two polynomials      */
/* Author                   : ASWIN BABU                 */
/* Date Written             : 23/03/2017                 */
/* Revision                 : 1                          */
*****/

/*****
/* PROGRAM:                                           */
/* coeff                   : To store the coefficient    */
/* pow                     : To store the power         */
/* ch                      : To store each element      */
/* poly1                   : To store the first polynomial */
/* poly2                   : To store the second polynomial */

#include<stdio.h>
#include<stdlib.h>
typedef struct link {

int coeff;

int pow;

struct link * next;

} my_poly;

void my_create_poly(my_poly **);

void my_show_poly(my_poly *);

void my_add_poly(my_poly **, my_poly *, my_poly *);

int main(void) {

int ch;

do {

my_poly * poly1, * poly2, * poly3;

printf("\nCreate 1st expression\n");

my_create_poly(&poly1);

printf("\nStored the 1st expression");

```

```

my_show_poly(poly1);
printf("\nCreate 2nd expression\n");
my_create_poly(&poly2);
printf("\nStored the 2nd expression");
my_show_poly(poly2);
my_add_poly(&poly3, poly1, poly2);
my_show_poly(poly3);
printf("\nAdd two more expressions? (Y = 1/N = 0): ");
scanf("%d", &ch);
} while (ch);
return 0;
}
void my_create_poly(my_poly ** node) {
    int flag;    int coeff, pow;
    my_poly * tmp_node    tmp_node = (my_poly *)
malloc(sizeof(my_poly));
    *node = tmp_node;
    do {
        printf("\nEnter Coeff:");
        scanf("%d", &coeff);
        tmp_node->coeff = coeff;
        printf("\nEnter Pow:");
        scanf("%d", &pow);
        tmp_node->pow = pow;
        tmp_node->next = NULL;
        printf("\nContinue adding more terms to the polynomial list?(Y =
1/N = 0): ");
        scanf("%d", &flag);

```

```

    if(flag) {
        tmp_node->next = (my_poly *) malloc(sizeof(my_poly));
        tmp_node = tmp_node->next;
        tmp_node->next = NULL;
    }
} while (flag);
}

void my_show_poly(my_poly * node) {
    printf("\n\nThe polynomial expression is:\n\n");
    while(node != NULL) {
        printf("%dx^%d", node->coeff, node->pow);
        node = node->next;
        if(node != NULL)
            printf(" + ");
    }
}

void my_add_poly(my_poly ** result, my_poly * poly1, my_poly *
poly2) {
    my_poly * tmp_node;
    tmp_node = (my_poly *) malloc(sizeof(my_poly));
    tmp_node->next = NULL;
    *result = tmp_node;
    while(poly1 && poly2) {
        if (poly1->pow > poly2->pow) {
            tmp_node->pow = poly1->pow;
            tmp_node->coeff = poly1->coeff;
            poly1 = poly1->next;

```

```

}
else if (poly1->pow < poly2->pow) {
    tmp_node->pow = poly2->pow;
    tmp_node->coeff = poly2->coeff;
    poly2 = poly2->next;
}
else {
    tmp_node->pow = poly1->pow;
    tmp_node->coeff = poly1->coeff + poly2->coeff;
    poly1 = poly1->next;
    poly2 = poly2->next;
}

if(poly1 && poly2) {
    tmp_node->next = (my_poly *) malloc(sizeof(my_poly));
    tmp_node = tmp_node->next;
    tmp_node->next = NULL;
}

}

while(poly1 || poly2) {
    tmp_node->next = (my_poly *) malloc(sizeof(my_poly));
    tmp_node = tmp_node->next;
    tmp_node->next = NULL;

    if(poly1) {
        tmp_node->pow = poly1->pow;

```

```
    tmp_node->coeff = poly1->coeff;
    poly1 = poly1->next;
}
if(poly2) {
    tmp_node->pow = poly2->pow;
    tmp_node->coeff = poly2->coeff;
    poly2 = poly2->next;
}
printf("\nAddition Complete");}

/*****/
```

```
/*  
/*
```

OUTPUT:

Create 1st expression

Enter Coeff:1

Enter Pow:1

Continue adding more terms to the polynomial list?(Y = 1/N = 0): 1

Enter Coeff:5

Enter Pow:2

Continue adding more terms to the polynomial list?(Y = 1/N = 0): 1

Enter Coeff:19

Enter Pow:3

Continue adding more terms to the polynomial list?(Y = 1/N = 0): 0

Stored the 1st expression

The polynomial expression is:

$$1x^1 + 5x^2 + 19x^3$$

Create 2nd expression

Enter Coeff:2

Enter Pow:1

Continue adding more terms to the polynomial list?(Y = 1/N = 0): 1

Enter Coeff:4

Enter Pow:2

Continue adding more terms to the polynomial list?(Y = 1/N = 0): 1

Enter Coeff:5

Enter Pow:3

Continue adding more terms to the polynomial list?(Y = 1/N = 0): 1

Enter Coeff:16

Enter Pow:4

Continue adding more terms to the polynomial list?(Y = 1/N = 0): 1

Enter Coeff:56

Enter Pow:5

Continue adding more terms to the polynomial list?(Y = 1/N = 0): 0

Stored the 2nd expression

The polynomial expression is:

$$2x^1 + 4x^2 + 5x^3 + 16x^4 + 56x^5$$

Addition Complete

The polynomial expression is:

$$3x^1 + 9x^2 + 24x^3 + 16x^4 + 56x^5 */$$

/******
/