Week-6 UE20CS207 DSLAB

- Name : P K Navin Shrinivas
- SRN : PES2UG20CS237
- Section : D
- Batch : 2

Lab assignments 1 : Jhosephus problem

Code :

main.c

```c
#include <stdio.h>
#include <stdlib.h>
typedef struct node
{
    int data;
    struct node *link;
} node;

void insert(node **s, int id)
{
    node *temp = (node *)malloc(sizeof(node));
    temp->data = id;
    temp->link = NULL;
    if (*s == NULL)
    {
        *s = temp;
        temp->link = *s;
    }

    else
    {
        node *d = (*s)->link;
        while (d->link != *s)
        {
            d = d->link;
        }
        d->link = temp;
        temp->link = *s;
    }
}

void display(node *s)
{
    node *temp = s;
    do
    {
        printf(" This is the position that will live on the queue %d\t", s->data);
        s = s->link;
    } while (s != temp);
```

```
    printf("\n");
}

void josephus(node **s, int k)
{
    node *temp = *s;
    node *d = NULL;
    int i;
    while (temp->link != temp)
    {
        for (i = 0; i < k - 1; i++)
        {
            temp = temp->link;
        }
        d = temp->link;
        temp->link = d->link;
        free(d);
        d = NULL;
        temp = temp->link;
    }
    *s = temp;
}

void main()
{
    node *n = NULL;
    int i, k, num;
    printf("Enter size of the queue: ");
    scanf("%d", &num);
    printf("Enter value of k: ");
    scanf("%d", &k);
    for (i = 0; i < num; i++)
    {
        insert(&n, i + 1);
    }
    josephus(&n, k - 1);
    display(n);
}
```

Lab Assignment 2 : Stack Queue Array problem

Code :

main.c

```
#include "1_1.h"
#include <stdio.h>

int main()
{
    int queue[QUEUESIZE];
    int top=-1,front=-1;
```

```
    struct stack* st=(struct stack*)malloc(sizeof(struct stack));
    st->top=-1;

    int n;
    int a[(n/2)+1];
    printf("Enter total number of elements : ");
    scanf("%d" , &n);
    for(int i=0;i<n;i++){
        int c;
        scanf("%d",&c);
        pushe(st,c);
    }
    for(int i=0;i<n;i++){
        if(st->top != -1){
            queuepush(queue, &top, &front, stackpeek(st));
            pope(st);
        }
    }
    int dummy = 0;
    for(int i=0;i<n;i++){
        if(top != -1){
            a[dummy] = queuepeek(queue,&top,&front);
            queuepop(queue,&top,&front);
            queuepop(queue,&top,&front)  ;
            dummy++;
        }
    }

    for(int i=0;i<(n/2)+1;i++)
        printf("%d " , a[i]);

}
```

1_1.h

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<math.h>
#include<stdbool.h>

#define STACKSIZE 100

struct stack{
    int top;
    int data[STACKSIZE];
};

void pushe(struct stack* st , int d);
void pope(struct stack* st);
void displaystack(struct stack* st);
int stackpeek(struct stack* st);
```

```
#define QUEUESIZE 100

void queuepush(int* queue , int* top , int* front,int e);
void queuepop(int* queue,int* top , int* front);
int queuepeek(int* queue , int* top , int* front);
void queuedisplay(int* queue , int* top, int* front);
```

1_1.c

```
#include"1_1.h"

void pushe(struct stack* st , int d)
{
    if(st->top == STACKSIZE-1)
    {
        return;
    }
    else{
        (st->top)++;
        st->data[st->top]=d;
        return;
    }
}

void pope(struct stack* st)
{
    if(st->top==-1)
    {
        return;
    }
    else{
        int e=st->data[st->top];
        (st->top)--;
        return;
    }
}

int stackpeek(struct stack* st){
    return st->data[st->top];
}


void queuepush(int* queue , int* top , int* front,int e)
{
    if(*top == QUEUESIZE-1)
    {
        return;
    }
    else if(*top==-1 && *front==-1)
    {
        *top=0;
        *front=0;
```

```
        *(queue+*top)=e;
        return;
    }
    else{
        *top=*top+1;
        *(queue+*top)=e;
        return;

    }
}

void queuepop(int* queue,int* top , int* front)
{
    if(*top==-1 && *front==-1)
    {
        return;
    }
    else if(*front == *top)
    {
        *top=-1;
        *front=-1;
        return;
    }
    else
    {
        *front=*front+1;
    }
}

int queuepeek(int* queue , int* top , int* front)
{
    return *(queue+*front) ;
}
```