CSA0389

DS

T. Sanhith

192311228

4/09/24

Program test-01

① array insertion/deletion :-

```
#include <stdio.h>
void insertion(int arr[], int size, int pos, int elem){
 for(int i=size; i<position; i++){
    arr[i] = arr[i-1];
    }
    arr[i+1] = element }
void delection(int arr[], int size, int POS){
    for(int i<position; i>size-1; i--){
     arr[i] = arr[i+1];
    }
int main(){
int arr[] = {1,2,3,4,5};
int size = sizeof(arr)/sizeof(arr[0]);
  printf("after insertion")
  insertion(arr, size, 2, 10);
    for(int i=0, i<size; i++){
        printf("%d ", arr[i]);
  Printf("i=0 after delection");
    delection(arr, size, 1);
            for(int i=0; i<size, i++){
  printf("%d", arr[i]);
    }
    return 0; }
```

4/9/24 Thread

output:

after insertion :- 1,2,10,3,4,6

delection :- 1,2,10,4,6

stack using array implementation:-

```
# include <stdio.h>
# define max
 struct stack{
    int arr[max];
     int top;
    }
  void initialize (struct stack *s){
     (*s) top=-1; }
    int isempty (struct stacks) {
      return s.top==-1;
int isfull (struct stacks){
     return s.top == max-1; }
  void push (struct stack*s, int value){
     if (isfull (*s)){
      printf("stack is full");
       return;
     (*s). arr[++(*s).top] = value;
     printf("%d pushed to stack, value);
      }
int peek(stack*s) {
     if (s*top==-1){
      printf("underflow);
     }
     return s*arr[s*top];
      }
```

```
void display (stack *s){
    for( int i, s->top; i>0; i--){
     printf("%d", s->top[i]);
int main(){
    stack s;
    initialize(*s);
    push(&s, 10);
    push(&s, 20);
    push(&s, 30);
printf("%d", pop(&s));
printf("top %d", peek(&s);
    display(&s);
return 0;
}
```

output :-

10 pushed
20 pushed
30 pushed
stack [10, 20, 30]
pop 10
stack [20, 30]

(3) Queue

```
# include <stdio.h>
# include <stdlib.h>
  # define max 1000
    type def struct{
        int front;
       int rear;
        int int;
    }
```

```
void initialize(queue *q){
    q->front = 0;
    q->rear = -1;
    q->size = 0;
}
void isempty(queue q){
    q.size = 0;
}
void intoll(queue *q){
    q->size = max;
}
void enqueue(queue q, int value){
    if(*full(a)){
        printf("overflow");
    }
    q->arr[q->rear] = value;
    q->size++;
}
void display(queue *q){
    for(int i=0; i<q->size; i++){
        printf("%d", q->q->front+i) %max;
    }
}
int main(){
    queue q;
    enqueue(&a, 10);
    enqueue(&a, 20);
    printf("%d", dequeue(&q));
    display(&q);
    return 0;
}
```

output :-
10, 15, 20, 30
deque element 10
15