DATA STRUCTURE

DATE:26/07/24

1)Array implementation of stack

Program:

```
#include <stdio.h>
#include <stdlib.h>
#include <limits.h>
 #define MAX 100
typedef struct {
   int items[MAX];
   int top;
} Stack;
 void initialize(Stack *s) {
   s->top = -1;
}
 int isFull(Stack *s) {
   return s->top == MAX - 1;
}
 int isEmpty(Stack *s) {
   return s->top == -1;
}
 void push(Stack *s, int item) {
   if (isFull(s)) {
      printf("Stack overflow\n");
      return;
   }
   s->items[++(s->top)] = item;
```

```
}

int pop(Stack *s) {
   if (isEmpty(s)) {
      printf("Stack underflow\n");
      return INT_MIN;    }
   return s->items[(s->top)--];
}
 int peek(Stack *s) {
   if (isEmpty(s)) {
      printf("Stack is empty\n");
      return INT_MIN;    }
   return s->items[s->top];
}
 void display(Stack *s) {
   if (isEmpty(s)) {
      printf("Stack is empty\n");
      return;
   }
   printf("Stack elements are:\n");
   for (int i = s->top; i >= 0; i--) {
      printf("%d\n", s->items[i]);
   }
}
 int main() {
   Stack s;
   initialize(&s);
  push(&s, 10);
```

```
    push(&s, 20);
    push(&s, 30);
 printf("Top element is %d\n", peek(&s));

    printf("Stack before popping:\n");
    display(&s);
    printf("Popped element is %d\n", pop(&s));
    printf("Stack after popping:\n");
    display(&s);
    return 0;
}
```

OUTPUT:

```
Top element is 30
Stack before popping:
Stack elements are:
30
20
10
Popped element is 30
Stack after popping:
Stack elements are:
20
10
```

2)Linked list implementation of stack

Program:

```
#include <stdio.h>
#include <stdlib.h>
```

```
typedef struct Node {
    int data;
    struct Node *next;
} Node;
typedef struct {
    Node *top;
} Stack;
void initialize(Stack *s) {
    s->top = NULL;
}
int isEmpty(Stack *s) {
    return s->top == NULL;
}
void push(Stack *s, int item) {
    Node *newNode = (Node *)malloc(sizeof(Node));
    if (newNode == NULL) {
        printf("Memory allocation failed\n");
        return;
    }
    newNode->data = item;
    newNode->next = s->top;
    s->top = newNode;
}
int pop(Stack *s) {
    if (isEmpty(s)) {
        printf("Stack underflow\n");
        return -1;     }
    Node *temp = s->top;
```

```
    int item = temp->data;
    s->top = s->top->next;
    free(temp);
    return item;
}
 int peek(Stack *s) {
    if (isEmpty(s)) {
        printf("Stack is empty\n");
        return -1;     }
    return s->top->data;
}
 void display(Stack *s) {
    if (isEmpty(s)) {
        printf("Stack is empty\n");
        return;
    }
    printf("Stack elements are:\n");
    Node *current = s->top;
    while (current != NULL) {
        printf("%d\n", current->data);
        current = current->next;
    }
}
 int main() {
    Stack s;
    initialize(&s);
 push(&s, 10);
    push(&s, 20);
```

```
    push(&s, 30);
   printf("Top element is %d\n", peek(&s));
    printf("Stack before popping:\n");
    display(&s);
    printf("Popped element is %d\n", pop(&s));
    printf("Stack after popping:\n");
    display(&s);
    return 0;
}
```

OUTPUT:

```
Top element is 60
Stack before popping:
Stack elements are:
60
50
40
Popped element is 60
Stack after popping:
Stack elements are:
50
40
```