23/12/23 Evaluation of Infix to Postfix

①
```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#define MAX 100
char stack[max];
char infix[max], postfix[max];
int top=-1;
void push(char);
char pop();
int isEmpty();
void inToPost();
void print();
int precedence(char);
int main()
{
  printf("Enter the infix expression");
  gets(infix);
  inToPost();
  print();
  return 0;
}
void inToPost()
{
  int i, j=0;
  char symbol, next;
  for(i=0; i<strlen(infix); i++)
  {
    symbol=infix[i];
    switch(symbol)
  {
    case '(':
      push(symbol);
```

```
case ')':
    while((next=pop()) != '(')
        postfix[j++] = next;
    break;
case '+':
case '-':
case '*':
case '/':
case '^':
    while(!isEmpty() && precedence(stack[top]) >= precedence(symbol))
        postfix[j++] = pop();
    push(symbol);
    break;
default:
    postfix[j++] = symbol;
}
}
while(!isEmpty())
    postfix[j++] = pop();
postfix[j] = '\0';
}

int precedence(char symbol)
{
    switch(symbol)
    {
    case '^':
        return 3;
    case '/':
    case '*':
        return 2;
    case '+':
    case '-':
        return 1;
    default:
        return 0;
```

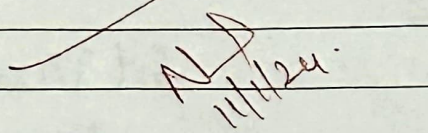
```
void fprint()
{
  int i=0;
  printf("The equivalent postfix expression is:");
  while(postfix[i])
  {
    printf("%c", postfix[i++]);
  }
  printf("\n");
}
void push(char c)
{
  if(top==MAX-1)
  {
    printf("Stack overflow");
    return;
  }
  top++;
  stack[top]=c;
}
char pop()
{
  char c;
  if(top==-1)
  {
    printf("Stack underflow");
    exit(1);
  }
  c=stack[top];
  top=top-1;
  return c;
}
```

```
int isEmpty()
{
  if (top==-1)
    return 1;
  else
    return 0;
}
```

Output:

Enter the infix expression: (a*b)+(c*d)

The equivalent postfix expression is: ab* cd* +



NLP
11/11/24

Queue

②
```
#include<stdio.h>
#include<stdlib.h>
#define max 6
int Q[MAX];
int front=-1;
int rear=-1;
void insert(int x){
  if(rear==MAX-1)
  {
    printf("Queue overflow");
    exit(1);
  }
  if(front==-1){
    front=rear=0;
  }
  else
  {
    rear++;
  }
  Q[rear]=x;
}
void delete(){
  if(front==-1)
  {
    printf("Queue underflow");
    exit(1);
  }
  printf("deleted element=%d\n",Q[front]);
  front++;
  if(front>rear)
  {
    front=rear=-1;
```

```
void display()
{
  if (front == -1)
  {
    printf("Queue underflow");
    exit(1);
  }
  int i;
  printf("front to rear: ");
  for(i=front; i<=rear; i++)
  {
    printf("%d", Q[i]);
  }
  printf("\n");
}
void main()
{
  int choice;
  int x;
  do{
    printf("Queue Operations");
    printf("1. Insertion");
    printf("2. deletion\n");
    printf("3. display\n");
    printf("4. exit\n");
    printf("enter the choice");
    scanf("%d", &choice);
    switch(choice)
  {
    case 1:
      printf("Enter the element to be inserted");
      scanf("%d", &x);
      insert(x);
      break;
```

```
case 3: display();
        break;
case 4: exit(1);
default: printf("Invalid choice try again");
}while(choice!=4);
}
```

Output:

```
Queue operation
1. insertion
2. deletion
3. display
4. exit
enter the choice 1
enter the element to be inserted 34

Queue operation
1. insertion
2. deletion
3. display
4. exit
enter the choice 2
enter the element to be deleted 34

Queue Operation
1. insertion
2. deletion
3. display
4. exit
enter the choice 4
```