Question -1:

#include<stdio.h>

int stack[100],choice,n,top,x,i;

void push(void);

void pop(void);

void display(void);

int main()

{

top=-1;

printf("\n Enter the size of STACK[MAX=100]:");

scanf("%d",&n);

printf("\nSTACK OPERATIONS USING ARRAY");

printf("\n1.PUSH\n2.POP\n3.DISPLAY\n4.EXIT");

do

{

printf("\n Enter the Choice:");

scanf("%d",&choice);

switch(choice)

{

case 1:

{

push();

break;

}

case 2:

{

pop();

break;

}

case 3:

{

display();

break;

}

case 4:

{

printf("\nEXIT POINT ");

break;

}

default:

{

printf ("\nPlease Enter a Valid Choice(1/2/3/4)");

}

}

}

while(choice!=4);

return 0;

}

void push()

{

if(top>=n-1)

{

printf("\nSTACK is over flow");

}

else

{

printf(" Enter a value to be pushed:");

scanf("%d",&x);

top++;

stack[top]=x;

}

}

void pop()

{

if(top<=-1)

{

printf("\nStack is under flow");

}

else

{

printf("\nThe popped elements is %d",stack[top]);

top--;

}

}

void display()

{

if(top>=0)

{

printf("\n The elements in STACK \n");

for(i=top; i>=0; i--)

printf("\n%d",stack[i]);

printf("\n Press Next Choice");

}

else

{

printf("\n The STACK is empty");

}

}

Question -2:

#include <limits.h>

#include <stdio.h>

#include <stdlib.h>

char \*arrayStack = NULL;

int top = -1;

totalSize = -1;

int isEmpty()

{

if (totalSize == -1 || arrayStack == NULL)

return -1;

return top == -1;

}

int isFull()

{

if (totalSize == -1 || arrayStack == NULL)

return -1;

return top == totalSize - 1;

}

char peek()

{

if (isEmpty())

return -1;

return arrayStack[top];

}

char pop()

{

if (isEmpty())

return -1;

char ch = arrayStack[top];

top--;

return (ch);

}

void push(char element)

{

if (isFull())

printf("Stack is already Full.");

else

arrayStack[++top] = element;

}

int precedence(char ch)

{

switch (ch)

{

case '+':

case '-':

return 1;

case '\*':

case '/':

return 2;

case '^':

return 3;

}

return -1;

}

int isOperand(char element)

{

if (element >= 'A' && element <= 'Z' || element >= 'a' && element <= 'z' || element >= '1' && element <= '9')

return 1;

else

return 0;

}

void infixToPostfix(char \*expressionArray)

{

printf("Corresponding Postfix Expression: ");

int currIndex = 0;

while (expressionArray[currIndex] != '\0')

{

if (isOperand(expressionArray[currIndex]))

printf("%c", expressionArray[currIndex]);

else if (expressionArray[currIndex] == '(')

push(expressionArray[currIndex]);

else if (expressionArray[currIndex] == ')')

{

while (peek() != '(')

printf("%c", pop());

pop();

}

else

{

while (precedence(peek()) >= precedence(expressionArray[currIndex]))

printf("%c", pop());

push(expressionArray[currIndex]);

}

currIndex++;

}

while (top != -1)

{

printf("%c", pop());

}

}

int main()

{

char expressionArray[40];

totalSize = 40;

arrayStack = (char \*)malloc(sizeof(char) \* 40);

printf("Enter a Infix Expression: ");

scanf("%s", expressionArray);

printf("Entered Infix Expression: %s\n", expressionArray);

infixToPostfix(expressionArray);

return 0;

}

Question -3:

#include <stdio.h>

#include <string.h>

int addsub();

int muldiv();

int term();

char input[101];

int pos = 0;

int term(){

int n = 0;

if(input[pos] == '('){

pos++;

n = addsub();

if(input[pos] == ')'){

pos++;

return n;

}

}else{

while('0' <= input[pos] && input[pos] <= '9'){

n = n\*10 + (input[pos] - '0');

pos++;

}

}

return n;

}

int muldiv(){

int first,second;

first = term();

for(;;){

if(input[pos] == '\*'){

pos++;

second = term();

first \*= second;

}else if(input[pos] == '/'){

pos++;

second = term();

first /= second;

}else{

return first;

}

}

}

int addsub(){

int first,second;

first = muldiv();

for(;;){

if(input[pos] == '+'){

pos++;

second = muldiv();

first += second;

}else if(input[pos] == '-'){

pos++;

second = muldiv();

first -= second;

}else{

return first;

}

}

}

int main(){

int n,i,j;

printf("Input an expression using +, -, \*, / operators:\n");

scanf("%s",input);

printf("%d\n",addsub());

return 0;

}

Question -4:

#include <stdio.h>

void move(int num\_disks, int source, int destination, int auxiliary) {

if (num\_disks == 1) {

printf("Move disk 1 from rod %d to rod %d\n", source, destination);

return;

}

move(num\_disks - 1, source, auxiliary, destination);

printf("Move disk %d from rod %d to rod %d\n", num\_disks, source, destination);

move(num\_disks - 1, auxiliary, destination, source);

}

int main() {

int num\_disks = 4;

move(num\_disks, 1, 3, 2);

return 0;

}