//IMPLEMENTATION OF STACK USING ARRAY//

#include<stdio.h>

#define MAX 100

int stack[MAX],choice,value,i,n;

int top=-1;

int is\_full()

{

if(top == MAX)

{

return true;

}

else

{

return false;

}

}

int is\_empty()

{

if(top != MAX)

{

return true;

}

else

{

return false;

}

}

void push()

{

if(top>=n-1)

{

printf("\n stack overflow");

}

else

{

printf("Enter the value to push:");

scanf("%d",&value);

top++;

stack[top] = value;

}

}

void pop()

{

if(top<=-1)

{

printf("\n stack underflow");

}

else

{

printf("Popped value=%d",stack[top]);

top--;

}

}

int peek()

{

printf("The peek value is:%d",stack[top]);

return 0;

}

void display()

{

if(top>=0)

{

printf("\n The elements in the stack are:\n");

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

{

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

}

printf("\npress the next choice");

}

else

{

printf("\nThe stack is empty");

}

}

int main()

{

printf("Enter the size of the stack:");

scanf("%d",&n);

printf("\n STACK OPERATIONS USING ARRAY");

printf("\n--------------------------------");

printf("\n 1.push\n 2.pop\n 3.peek\n 4.display\n 5.is\_full\n 6.is\_empty");

do

{

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

scanf("%d",&choice);

switch(choice)

{

case 1:

{

push();

break;

}

case 2:

{

pop();

break;

}

case 3:

{

peek();

break;

}

case 4:

{

display();

break;

}

case 5:

{

is\_full();

break;

}

case 6:

{

is\_empty;

break;

}

default:

{

printf ("\n\t Please Enter a Valid Choice(1/2/3/4/5/6)");

}

}

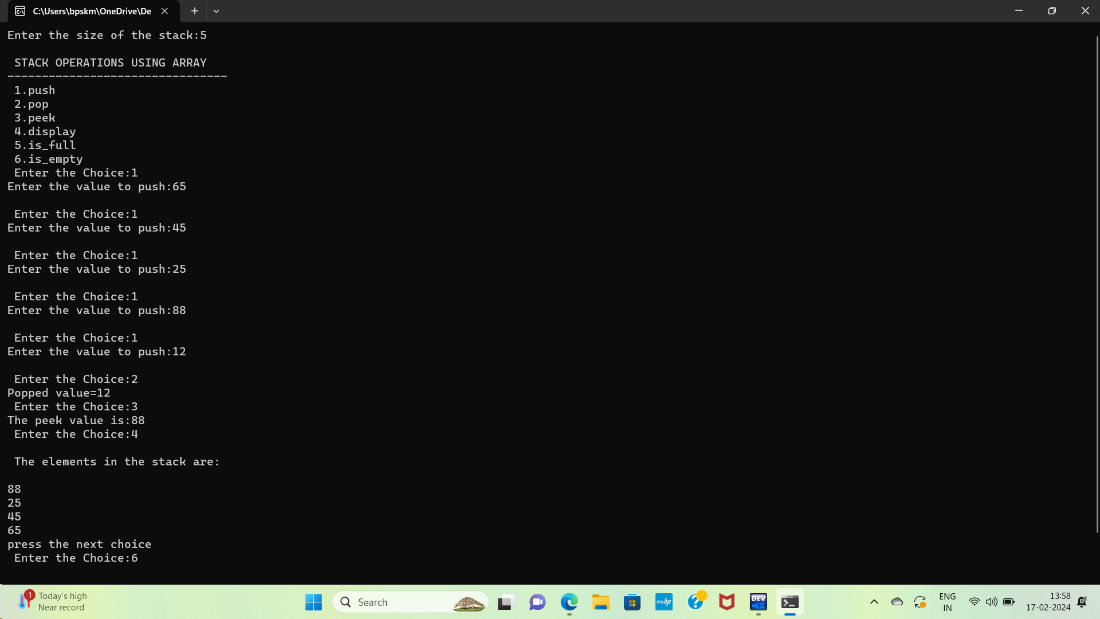
}

while(choice!=6);

return 0;

}

OUTPUT:



//CONVERSION OF INFIX TO POSTFIX EXPRESSION//

#include<stdio.h>

#include<ctype.h>

char stack[100];

int top = -1;

void push(char x)

{

stack[++top] = x;

}

char pop()

{

if(top == -1)

return -1;

else

return stack[top--];

}

int priority(char x)

{

if(x == '(')

return 0;

if(x == '+' || x == '-')

return 1;

if(x == '\*' || x == '/')

return 2;

if(x = '^')

return 3;

return 0;

}

int main()

{

char exp[100];

char \*e, x;

printf("Enter the expression : ");

scanf("%s",exp);

printf("\n");

e = exp;

while(\*e != '\0')

{

if(isalnum(\*e))

printf("%c ",\*e);

else if(\*e == '(')

push(\*e);

else if(\*e == ')')

{

while((x = pop()) != '(')

printf("%c ", x);

}

else

{

while(priority(stack[top]) >= priority(\*e))

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

push(\*e);

}

e++;

}

while(top != -1)

{

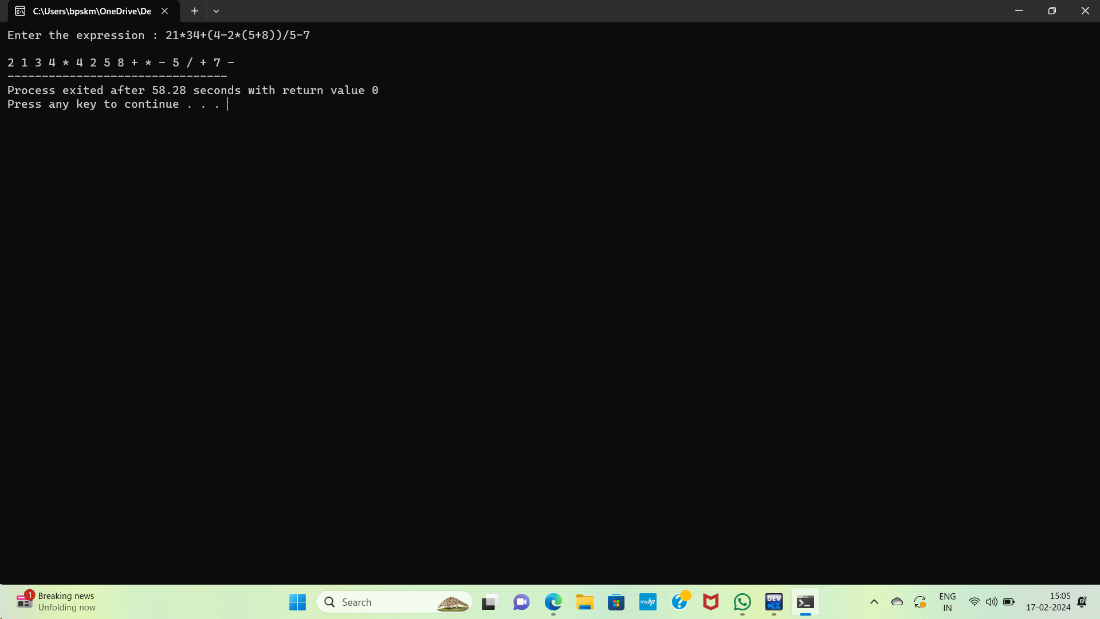
printf("%c ",pop());

}

return 0;

}

OUTPUT:



//EVALUATE THE POSTFIX EXPRESSION//

#include<stdio.h>

#include<ctype.h>

int stack[20];

int top=-1;

void push(int x)

{

stack[++top] = x;

}

int pop()

{

return stack[top--];

}

int main()

{

char exp[20];

char \*e;

int n1,n2,n3,num;

printf("Enter the expression::");

scanf("%s", exp);

e = exp;

while(\*e != '\0')

{

if(isdigit(\*e))

{

num = \*e-48;

push(num);

}

else

{

n1 = pop();

n2 = pop();

switch(\*e)

{

case '+':

{

n3 = n1 + n2;

break;

}

case '-':

{

n3 = n1 - n2;

break;

}

case '\*':

{

n3 = n1 \* n2;

break;

}

case '/':

{

n3 = n2 / n1;

break;

}

}

push(n3);

}

e++;

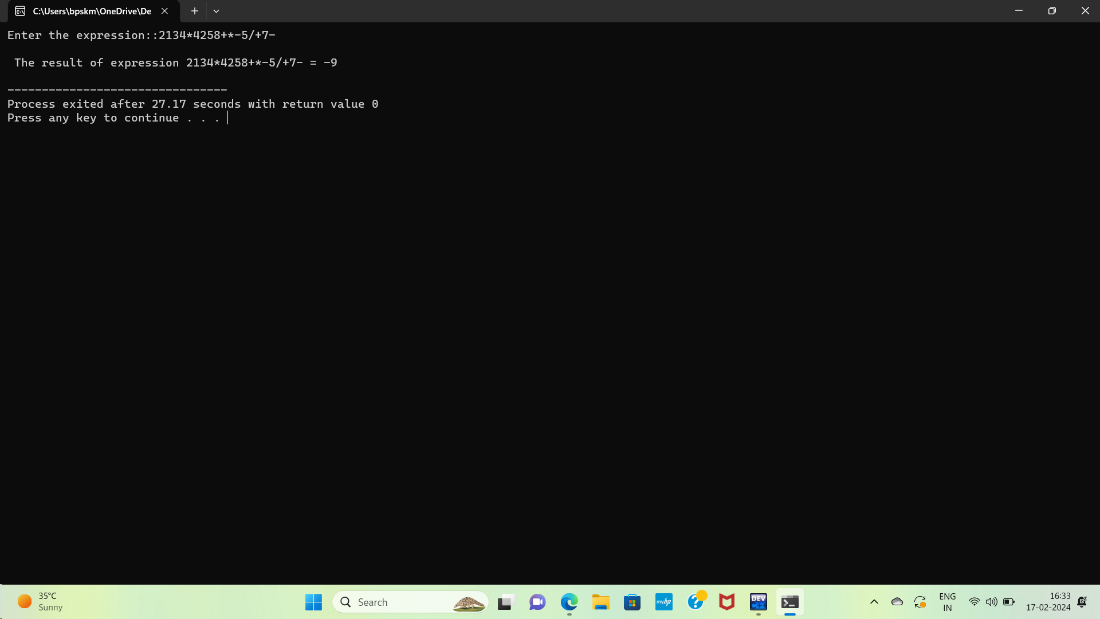
}

printf("\n The result of expression %s = %d\n",exp,pop());

return 0;

}

OUTPUT:



//TOWER OF HANOI//

#include<stdio.h>

void towerofHanoi(int n, char A,char B,char C)

{

if(n == 1)

{

printf("\n move disk 1 from rod %c to rod %c", A, C);

return ;

}

towerofHanoi(n - 1, A,B,C);

printf("\n move disk %d from rod %c to rod %c", n , A,C);

towerofHanoi(n-1,B,C,A);

}

int main()

{

int n = 4;

towerofHanoi(n,'A','C','B');

return 0;

}

OUTPUT:

