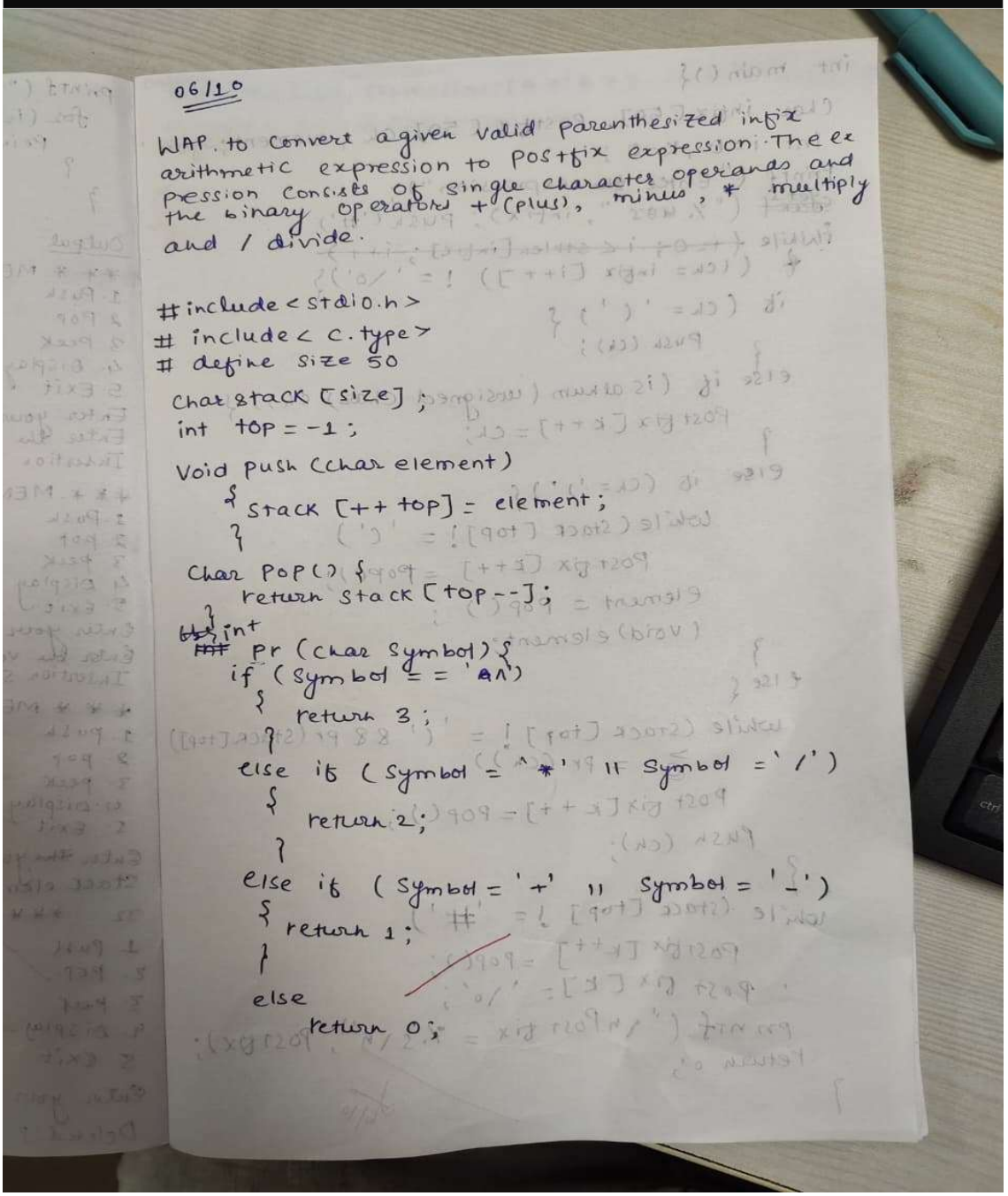
```
Enter the Infix expression: A+(B*C-(D/E^F)*G)*H

Postfix = ABC*DEF^/G*-H*+

Process returned 0 (0x0)   execution time : 55.971 s
Press any key to continue.
```



06/10

WAP to convert a given valid parenthesized infix arithmetic expression to postfix expression. The expression consists of single character operands and the binary operators + (plus), - (minus), * (multiply) and / (divide).

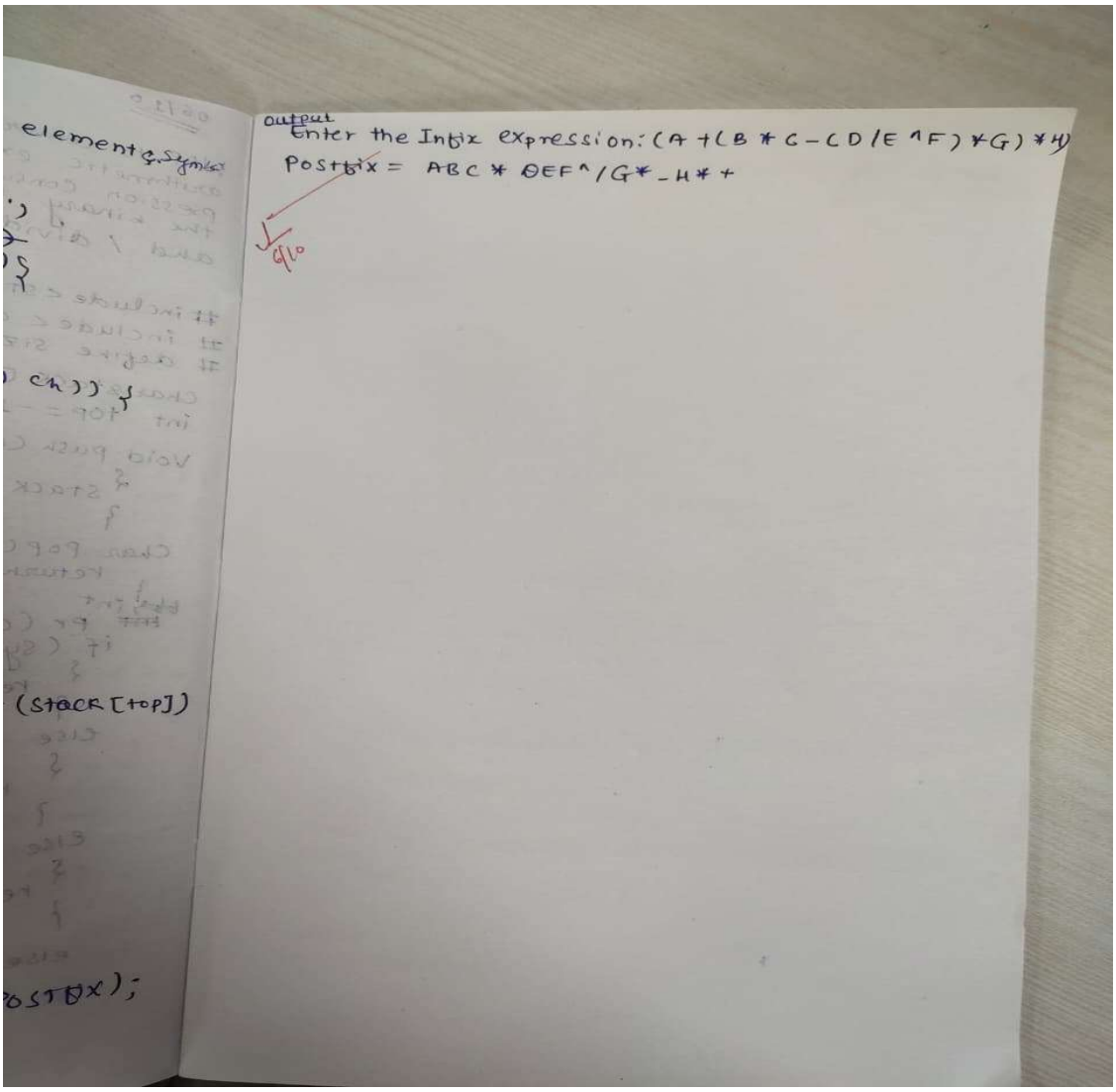
```
#include<stdio.h>
#include<ctype.h>
#define size 50
char stack[size];
int top = -1;
void push(char element)
{
    stack[++top] = element;
}
char pop(){
    return stack[top--];
}
int pr(char symbol){
    if(symbol == '^')
    {
        return 3;
    }
    else if(symbol == '*' || symbol == '/')
    {
        return 2;
    }
    else if(symbol == '+' || symbol == '-')
    {
        return 1;
    }
    else
        return 0;
```

```
int main(){
 Char infix[50], postfix[50], ch, element;
 int i=0, k=0;
Printf ("Enter the expression");
 scanf ("%s", infix); push('#');
while (ch= infix[i++]) != '\0'){
   if (ch= '(') {
       Push (ch);
   }
   else if (isalnum (unsigned char) ch)) {
       Postfix[k++] = ch;
   }
   else if (ch= ')') {
       while (stack[top]!= '(')
           Postfix[k++] = pop();
       element = pop();
       (void) element;
   }
   else {
    while (stack[top] != '(' && pr(stack[top])
          >= pr(ch))
       Postfix[k++] = pop();
     push (ch);
   }
while (stack[top] != '#')
     Postfix[k++] = pop();
   Postfix[k] = '\0';
   printf ("\n Postfix = %s\n", postfix);
   return 0;
}
```

8/10

output
Enter the In
Postfix = A

9/10


element
(stack[top])
output
Enter the Infix expression: (A+(B*C-CD/E^F)*G)*H)
Postfix = ABC*DEF^/G*-H*+