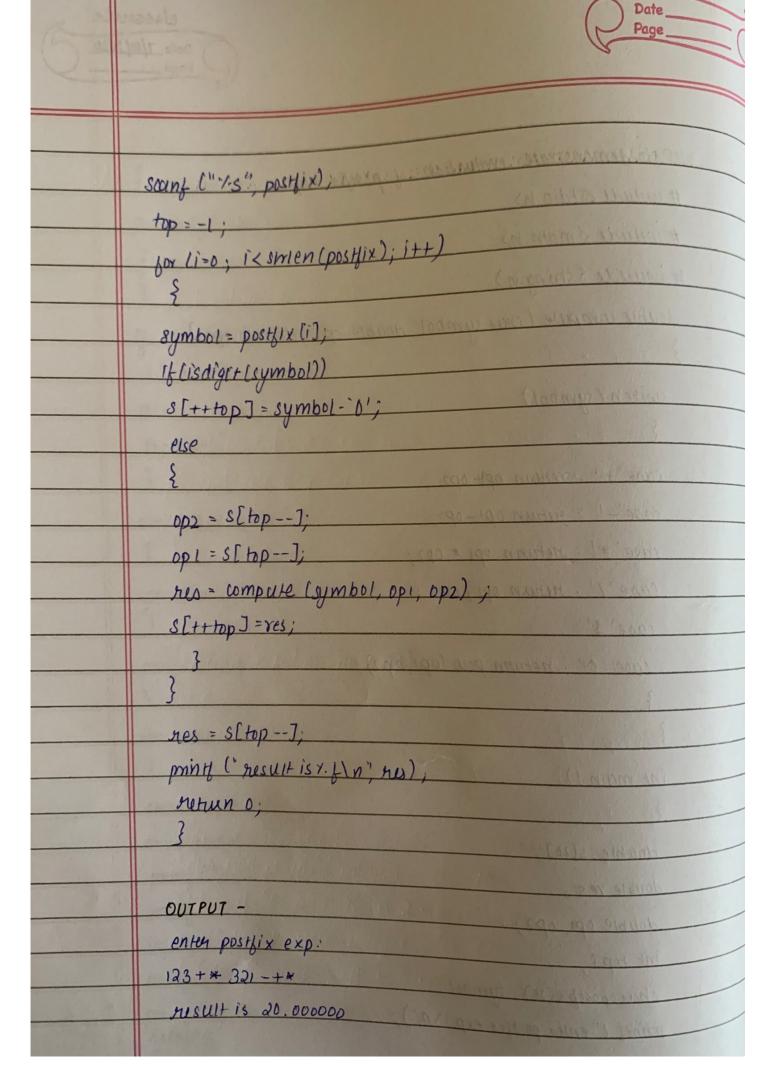
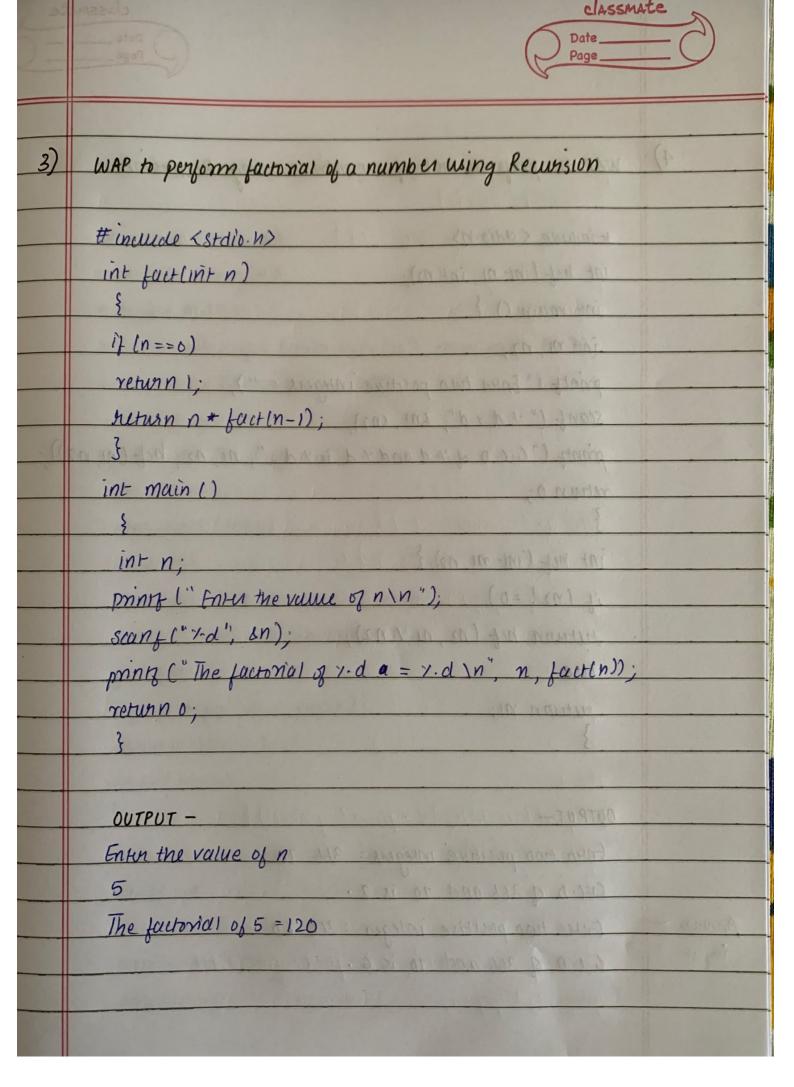
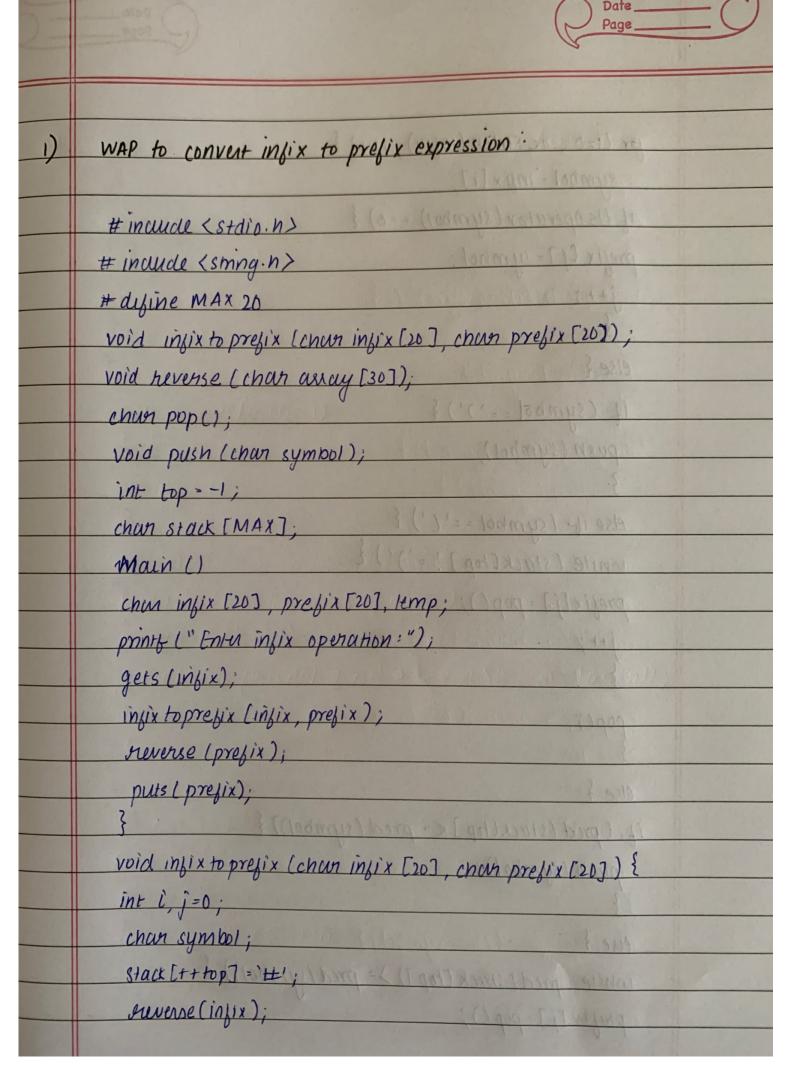


```
WAP to demonstrate evaluation of postfix exp.
# include (Stdio.n)
 # include < math. n>
# include (sming.n)
double compute (chan symbol, double op 1, double op2)
 switch (symbol)
  case + 1: return opt op2;
  case'-1: gutun op1-0P2;
  case '*1: neturn op1 * op2;
  case '/': return op! /op2; and lod myst summer and
  (ase'$':
   case 'A': netwin pow lop1, op2);
 int main ()
  double s[20];
 double res;
  double opl, op2;
  int topi;
 chan postfix [20], symbol;
 prints ("enth postlix exp =\n");
```





)	
	using Recursion
4)	WAP to perform GCD of 2 numbers using Recursion
	Finula Kardina
	#include <stdio.n></stdio.n>
	int hef (int n1, int n2);
	int main () {
	into its cause here precitive integers: ");
	count (" 1.1 1.1" LNI LNZ);
	print ("G.C.D of y-d and y-d is y-d.", n1, n2, het (n1, n2));
	retuno;
	3
	int net (int-n1, n2) {
	if [nz! =0) ("NYN B supplement that I find
	rutunn hef (n2, n1 1. n2);
	cheeses in my bis a bir o sometime ant is min-
	netwon ni;
	3
	OUTPUT -
	Enter two positive integers: 366 70
	01-C p of 066 and 70 u 2.
	Enter two positive integer : 366 60
	a.c.D & 366 and 70 is 6.
	A CONTRACTOR OF THE PARTY OF TH



```
Date
for (i=0; i < smen (infix); i++) {
  symbol = insix [i];
 if (is Operator (symbol) == 6) {
 prefix GJ = symbol;
  else {
  if (symbol == )) }
  pusn (symbol);
  else if (symbol = = '(') {
while (stack(top]!=')') {
  prefix[j] = pop();
  pop();
 else &
if (pred(stack(top) <= pred(symbol)) {
  push (symbol);
while (pred[stack [top]) ) = pred(symbol)) {
pryly[j]=pop();
```

```
push (symbol);
While (stack [top]! = '#1) {
prefix (j) = pop ();
prefix [j] = \\0';
void reverse [char away [30]) {
intil;
char temp[100];
fox (i = smen(anay)-1, j=0; i+1!=0; ++j) }
temp [j] = anay[i];
 temp = 101;
 sripy (anay, temp);
 (non pop () {
 chae a;
```

```
a = Stack [top];
 top --;
return a;
void push (chur symbol) }
 top++;
Stack [top] = symbol;
int pred (cher symbol) {
 Switch (symbol) {
 case +1:
case '-1: Metun 2; break;
case '*':
 case 11: return 4; break;
ease '$ ':
case'n': return 6; break; [00] any some
case #1: 1+1 10= 1+11 10-1 1- (What hatel + 1) 10
 case 'C':
 case ') ' : return 1; break;
int isopenator (char symbol) {
switch (symbol) {
case'+1:
```

17. 14. 15. 15.