Practical:-1

Aim :- Wap to find ASCII value of a character.

Input :-

#include<stdio.h>

int main()

{

char A;

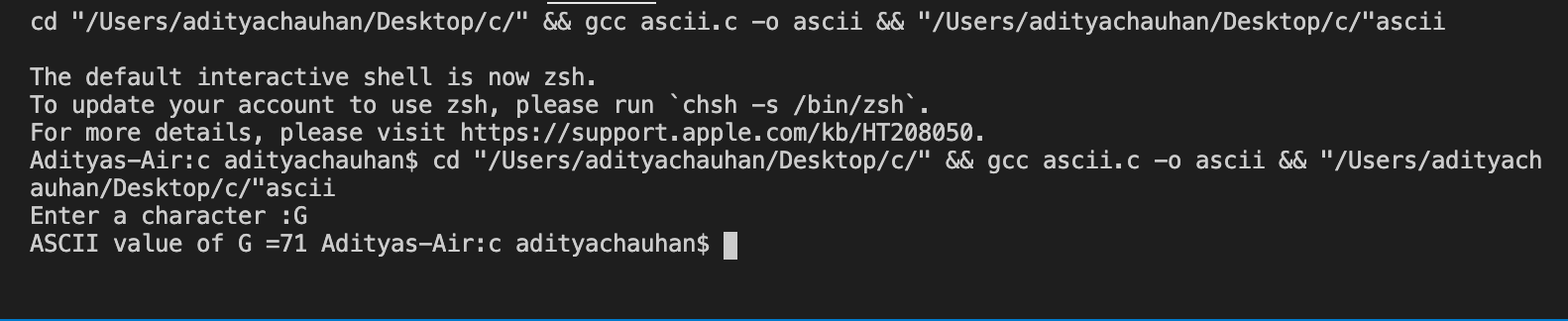
printf("Enter a character :");

scanf("%c",&A);

printf("ASCII value of %c =%d ",A,A);

return 0;

}

Output :-

Practical:-2

Aim :- Wap to swap two numbers without using third variable.

Input :-

#include<stdio.h>

int main()

{

int a,b;

printf("enter the value of a: ");

scanf("%d",&a);

printf("enter the value of b: ");

scanf("%d",&b);

printf("BEfore swap a=%d b=%d",a,b);

a=a\*b;

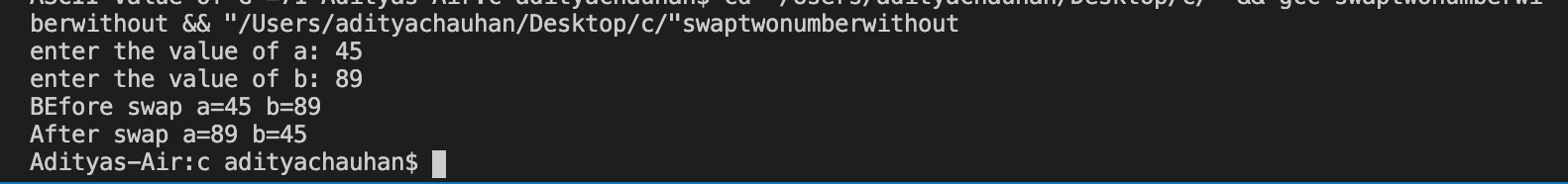
b=a/b;

a=a/b;

printf("\nAfter swap a=%d b=%d\n",a,b);

return 0;

}

Output :-

Practical:-3

Aim :-Wap to find roots of a quadratic equation.

Input :-

#include <math.h>

#include <stdio.h>

int main() {

double a, b, c, discriminant, root1, root2, realPart, imagPart;

printf("Enter coefficients a, b and c: ");

scanf("%lf %lf %lf", &a, &b, &c);

discriminant = b \* b - 4 \* a \* c;

if (discriminant > 0) {

root1 = (-b + sqrt(discriminant)) / (2 \* a);

root2 = (-b - sqrt(discriminant)) / (2 \* a);

printf("root1 = %.2lf and root2 = %.2lf", root1, root2);

}

else if (discriminant == 0) {

root1 = root2 = -b / (2 \* a);

printf("root1 = root2 = %.2lf;", root1);

}

else {

realPart = -b / (2 \* a);

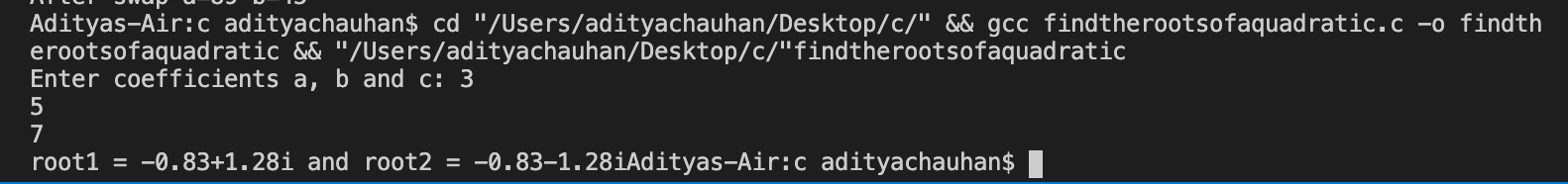
imagPart = sqrt(-discriminant) / (2 \* a);

printf("root1 = %.2lf+%.2lfi and root2 = %.2f-%.2fi", realPart, imagPart, realPart, imagPart);

}

return 0;

}

Output :-

Practical:-4

Aim :-Wap to find Factorial of a number.

Input :-

#include<stdio.h>

long double number(int n);

int main ()

{

int n;

printf("enter the integer:");

scanf("%d",&n);

printf("Factorial of %d = %Lf",n,number(n));

return 0;

}

long double number(int n)

{

if (n>=1)

{

return n\*number(n-1);

}

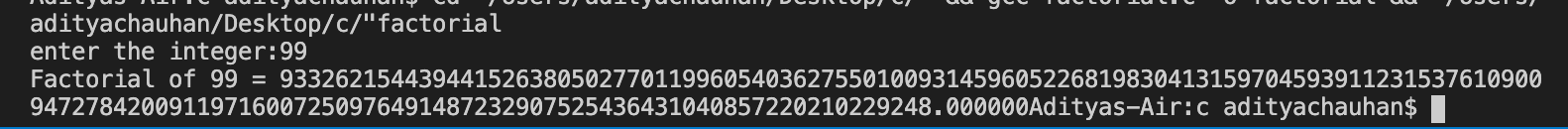
else

{

return 1;

}

}

Output :-

Practical:-5

Aim :-Wap to generate multiplication table.

Input :-

#include<stdio.h>

void tables(int);

int main()

{

int num;

printf("Enter a positive number\n");

scanf("%d", &num);

printf("\nMultiplication Table for %d is:\n", num);

tables(num);

return 0;

}

void tables(int num)

{

int count;

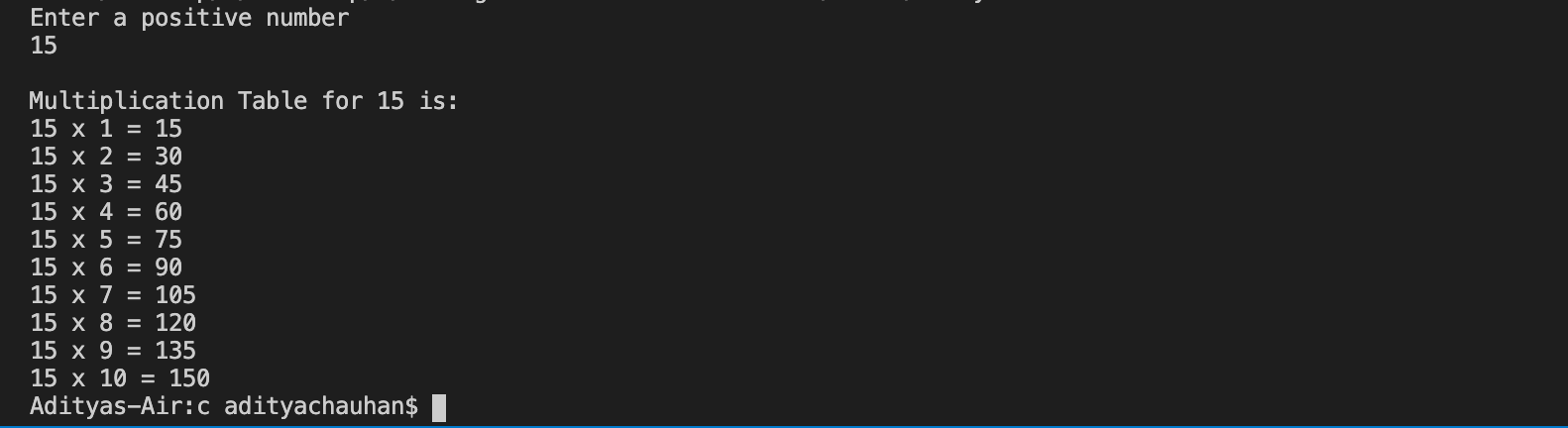
for(count = 1; count <= 10; count++)

{

printf("%d x %d = %d\n", num, count, num\*count);

}

}

Output :-

Practical:-6

Aim :-Wap to find GCD and LCM of two numbers.

Input :-

#include<stdio.h>

int main()

{

int num1, num2, i, gcd, lcm;

printf("Enter two numbers: ");

scanf("%d %d", &num1, &num2);

for(i=1; i<=num1 && i<=num2; i++)

{

if(num1%i==0 && num2%i==0)

gcd=i;

}

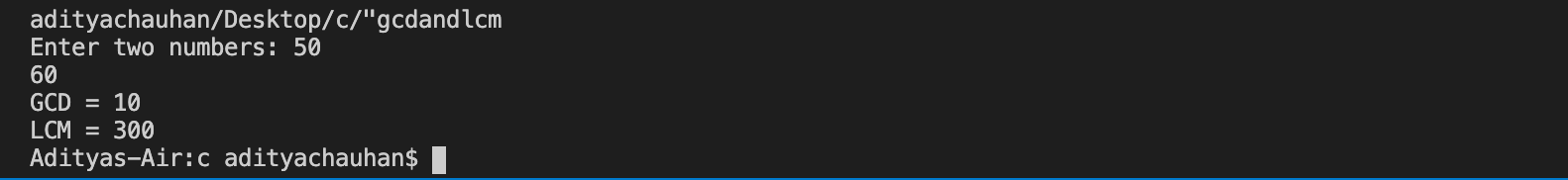
lcm=(num1\*num2)/gcd;

printf("GCD = %d\n",gcd);

printf("LCM = %d\n",lcm);

return 0;

}

Output :-

Practical:-7

Aim :-Wap to reverse a number.

Input :-

#include <stdio.h>

int main() {

int n, rev = 0, c;

printf("Enter an integer: ");

scanf("%d", &n);

while (n != 0) {

c = n % 10;

rev = rev \* 10 + c;

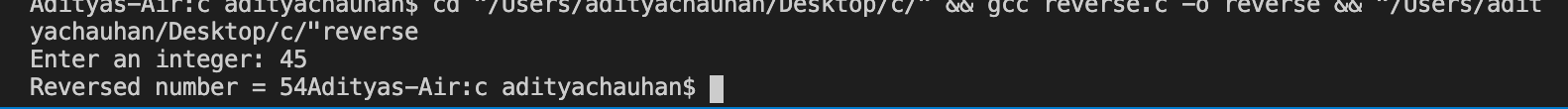
n /= 10;

}

printf("Reversed number = %d", rev);

return 0;

}

Output :-

Practical:-8

Aim :-Wap to check whether a number is palindrome or not.

Input :-

#include <stdio.h>

int main() {

int n, reversedN = 0, remainder, originalN;

printf("Enter an integer: ");

scanf("%d", &n);

originalN = n;

while (n != 0) {

remainder = n % 10;

reversedN = reversedN \* 10 + remainder;

n /= 10;

}

if (originalN == reversedN)

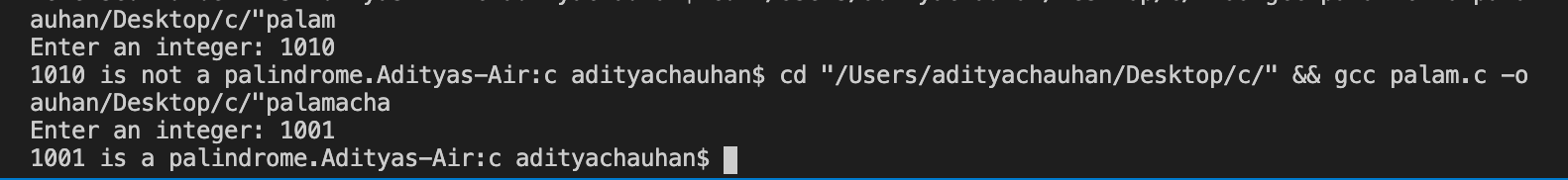
printf("%d is a palindrome.", originalN);

else

printf("%d is not a palindrome.", originalN);

return 0;

}

Output :-

Practical:-9

Aim :-Wap to calculator using switch case

Input :-

#include <stdio.h>

int main() {

char operator;

double first, second;

printf("Enter an operator (+, -, \*,/): ");

scanf("%c", &operator);

printf("Enter two operands: ");

scanf("%lf %lf", &first, &second);

switch (operator) {

case '+':

printf("%.1lf + %.1lf = %.1lf", first, second, first + second);

break;

case '-':

printf("%.1lf - %.1lf = %.1lf", first, second, first - second);

break;

case '\*':

printf("%.1lf \* %.1lf = %.1lf", first, second, first \* second);

break;

case '/':

printf("%.1lf / %.1lf = %.1lf", first, second, first / second);

break;

default:

printf("Error! operator is not correct");

}

return 0;

}

Output :-