**MAHARAJA SURAJMAL INSTITUTE**

**Affiliated to GGSIP University & NAAC ‘A’ grade accredited**



**DEPARTMENT OF COMPUTER**

**APPLICATIONS**

***C LANGUAGE***

**Practical File**

**Subject Code – BCA 171**

**SUBMITTED BY SUBMITTED TO**

**Aman Tripathi Mrs Minal Dhankar**

**05221202021 Asst. Professor. MSI**

**1st Sem, 2nd Shift**

Index

| **S.No.** | **Content** | **Sign.** |
| --- | --- | --- |
| 1. | WAP TO CALCULATE SUM AND PERCENTAGE OF A STUDENT IN FIVE SUBJECTS. |  |
| 2. | WAP TO CONVERT AND PRINT DISTANCE IN METRES, CENTIMETERS AND INCHES FROM KILOMETERS(INPUT). |  |
| 3. | WAP TO CONVERT FAHRENHEIT TO CELCIUS. |  |
| 4. | WAP TO READ PRINCIPAL AMOUNT, TIME AND RATE OF INTEREST FROM USER AND CALCULATE SIMPLE INTEREST. |  |
| 5. | WAP TO DISPLAY MINIMUM AND MAXIMUM RANGE OF DATATYPES. |  |
| 6. | WAP TO CONVERT DECIMAL NUMBERS INTO OCTAL AND HEXADECIMAL. |  |
| 7. | WAP TO SWAP THE TWO NUMBERS WITHOUT USING THE THIRD VARIABLE. |  |
| 8. | WAP TO SHOW SHIFT BITWISE OPERATOR. |  |
| 9. | WAP TO PRINT THE FIRST 10 SQUARE NUMBERS. |  |
| 10. | WAP TO CALCULATE THE FACTOR OF A NUMBER. |  |
| 11. | WAP TO FIND THE GREATEST OF THREE NUMBERS USING ‘NESTED IF’. |  |
| 12 | WAP TO FIND THE GREATEST OF THREE NUMBERS USING ‘CONDITIONAL OPERATOR’. |  |
| 13. | WAP TO CALCULATE THE RECIPROCAL OF ANY ‘N’ NUMBERS. |  |
| 14. | WAP TO GENERATE THE VARIOUS PYRAMIDS. |  |
| 15. | WAP TO REVERSE A DIGIT OF A GIVEN POSITIVE NUMBER USING DO-WHILE. |  |
| 16. | WAP TO CHECK WHETHER THE ENTERED CHARACTER IS LETTER, DIGIT OR NONE OF THESE. |  |
| 17. | WAP TO DEMONSTRATE HOW A FUNCTION SHOULD BE CALLED WITHIN ANOTHER FUNCTION. |  |
| 18. | WAP TO DETERMINE THE GREATEST OF THREE NUMBERS USING FUNCTIONS. |  |
| 19. | WAP TO CHECK WHETHER THE GIVEN NUMBER IS PRIME OR NOT USING FUNCTIONS. |  |
| 20. | WAP TO DETERMINE PRIME NUMBERS BETWEEN 1 TO 100. |  |
| 21. | WAP TO CALCULATE THE HCF OF TWO NUMBERS USING FUNCTIONS. |  |
| 22. | WAP TO SHOW FUNCTIONS OF MATRIX. |  |
| 23. | WAP TO SHOW THE FOLLOWING STRING FUNCTIONS |  |
| 24. | WAP TO DEMONSTRATE THE STRUCTURE: -   1. A BASIC PROGRAM OF STRUCTURE 2. ASSIGNMENT IN OPERATOR 3. STORAGE IN MEMORY |  |
| 25. | WAP TO SHOW HOW INITIALIZATION IS DONE IN AN ARRAY OF STRUCTURES. |  |
| 26. | WAP TO SHOW A NESTED STRUCTURE. |  |
| 27. | WAP TO SHOW THE ADDRESS OF EACH VARIABLE IN UNION |  |
| 28. | WAP TO SHOW HOW TO INITIALIZE THE MEMBERS OF A UNION AND CHECK THEIR ADDRESS. |  |
| 29. | WAP TO DEMONSTRATE STORAGE CLASSES: -  • STATIC  • REGISTER  • AUTOMATIC |  |
| 30. | WAP TO OPEN AND CLOSE THE FILE. |  |
| 31. | WAP TO WRITE AND READ ANY TEXT INTO THE FILE BY USING THE FUNCTIONS fputc() AND fgetc(). |  |
| 32. | WAP TO INPUT A LINE INTO THE FILE AND DISPLAY ITS CONTENTS. |  |
| 33. | WAP TO SHOW THE USE OF fscanf() AND fprintf(). |  |
| 34. | WAP TO ILLUSTRATE ftell() AND fseek(). |  |
| 35. | WAP TO ILLUSTRATE THE REWIND() FUNCTION. |  |
| 36. | WAP TO ILLUSTRATE fwrite() AND fread(). |  |
| 37. | WAP TO DEMONSTRATE THE CONCEPT OF ENUM DATATYPE. |  |
| 38. | WAP TO DEMONSTRATE THE CONCEPT OF STRUCTURE WITHIN A FUNCTION: -  1. SINGLE VARIABLE  2. ENTIRE STRUCTURE |  |
| 39. | WAP TO DEMONSTRATE PASSING ARGUMENTS IN MAIN FUNCTIONS. |  |
|  |  |  |

Practical 1

Code:-

\*\*

\* C program to calculate total, average and percentage of five subjects

\*/

#include <stdio.h>

int main()

{

float eng, phy, chem, math, comp;

float total, average, percentage;

/\* Input marks of all five subjects \*/

printf("Enter marks of five subjects: \n");

scanf("%f%f%f%f%f", &eng, &phy, &chem, &math, &comp);

/\* Calculate total, average and percentage \*/

total = eng + phy + chem + math + comp;

average = total / 5.0;

percentage = (total / 500.0) \* 100;

/\* Print all results \*/

printf("Total marks = %.2f\n", total);

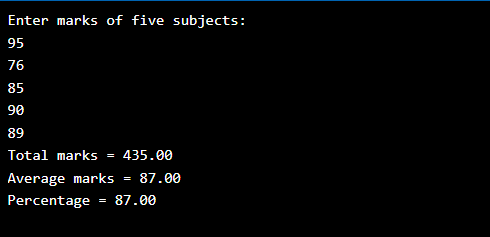
printf("Average marks = %.2f\n", average);

printf("Percentage = %.2f", percentage);

return 0;

}

Output:-



Practical 2

Code:-

/\* C Program to convert input distance in meter, feet, inches, centimeter \*/

#include <stdio.h>

#include <conio.h>

int main() {

int distance;

float meter, feet, inches, centimeter;

printf("Enter the distance [in Kilometers]: ");

scanf("%d", & distance);

meter = distance \* 1000;

feet = distance \* 3280.84;

inches = distance \* 39370.1;

centimeter = distance \* 100000;

printf("Meter = %f\n", meter);

printf("Feet = %f\n", feet);

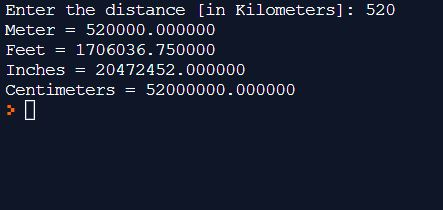
printf("Inches = %f\n", inches);

printf("Centimeters = %f\n", centimeter);

getch();

}

Output:-



Practical 3

Code:-

#include <stdio.h>

void main()

{

int fahrenheit = 0;

int celsius = 0;

printf("Enter the Temperature in Fahrenheit: ");

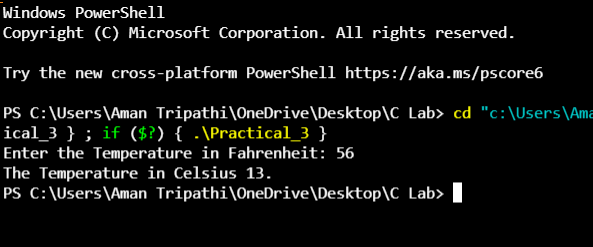
scanf("%d", &fahrenheit);

celsius = ((fahrenheit - 32) \* 5) / 9;

printf("The Temperature in Celsius %d.", celsius);

}

Output:-



Practical 4

Code:-

#include <stdio.h>

void main()

{

int principal, year;

float rate, simpleInterest;

for (int i = 0; i < 3; i++)

{ // for loop

printf("Enter the Value of Principle ,Year and rate\n");

scanf("%d%d%f", &principal, &year, &rate);

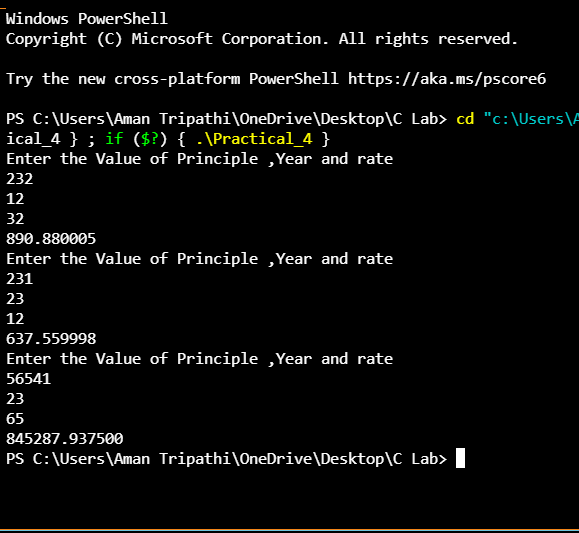
simpleInterest = (principal \* rate \* year) / 100;

printf("%f\n", simpleInterest);

}

}

Output:-



Practical 5

Code:-

/\*

\* C Program to Print the Range

\*/

#include <stdio.h>

#define SIZE(x) sizeof(x)\*8

void signed\_one(int);

void unsigned\_one(int);

void main()

{

printf("\nrange of int");

signed\_one(SIZE(int));

printf("\nrange of unsigned int");

unsigned\_one(SIZE(unsigned int));

printf("\nrange of char");

signed\_one(SIZE(char));

printf("\nrange of unsigned char");

unsigned\_one(SIZE(unsigned char));

printf("\nrange of short");

signed\_one(SIZE(short));

printf("\nrange of unsigned short");

unsigned\_one(SIZE(unsigned short));

}

/\* RETURNS THE RANGE SIGNED\*/

void signed\_one(int count)

{

int min, max, pro;

pro = 1;

while (count != 1)

{

pro = pro << 1;

count--;

}

min = ~pro;

min = min + 1;

max = pro - 1;

printf("\n%d to %d", min, max);

}

/\* RETURNS THE RANGE UNSIGNED \*/

void unsigned\_one(int count)

{

unsigned int min, max, pro = 1;

while (count != 0)

{

pro = pro << 1;

count--;

}

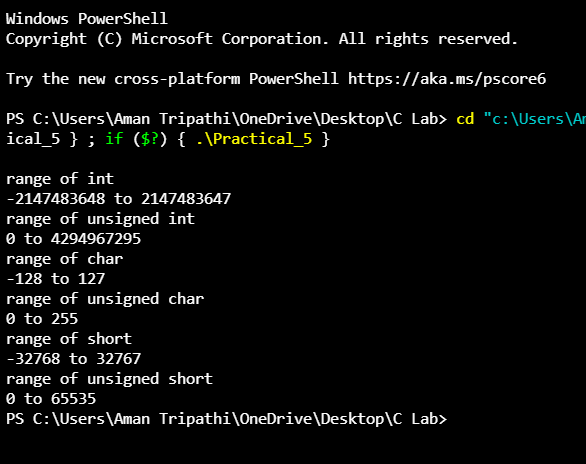
min = 0;

max = pro - 1;

printf("\n%u to %u", min, max);

}

Output:-



Practical 6

Code:-

/\* Program to convert a positive decimal number to Binary, Octal or Hexadecimal \*/

#include<stdio.h>

void convert(int, int);

int main()

{

int num;

printf("Enter a positive decimal number : ");

scanf("%d", &num);

printf("\nBinary number :: ");

convert(num, 2);

printf("\n");

printf("\nOctal number :: ");

convert(num, 8);

printf("\n");

printf("\nHexadecimal number :: ");

convert(num, 16);

printf("\n");

return 0;

}/\*End of main()\*/

void convert (int num, int base)

{

int rem = num%base;

if(num==0)

return;

convert(num/base, base);

if(rem < 10)

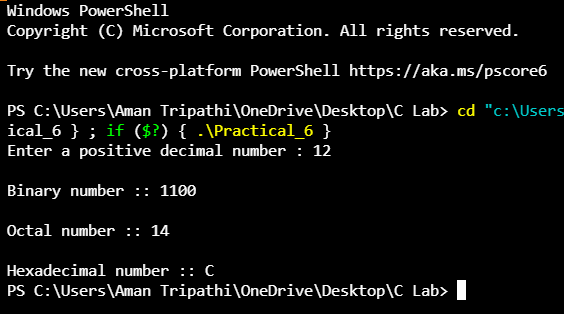
printf("%d", rem);

else

printf("%c", rem-10+'A' );

}/\*End of convert()\*/

Output:-



Practical 7

Code:-

#include <stdio.h>

// #include <stdlib.h>

int main()

{

int a = 10, b = 20;

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

a = a \* b; // a=200 (10\*20)

b = a / b; // b=10 (200/20)

a = a / b; // a=20 (200/10)

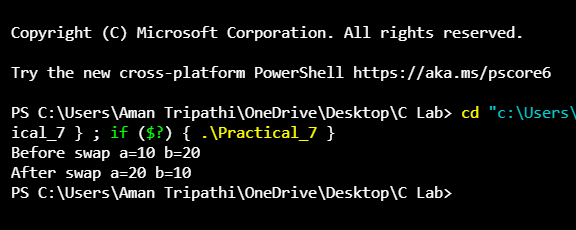
// system("cls");

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

return 0;

}

Output:-



Practical 8

Code:-

#include<stdio.h>

int main()

{

int var = 128;

printf("var/2 =%d \n",var>>1); //1 position to right

printf("var/4 =%d \n",var>>2); //2 position to right

printf("var/8 =%d \n",var>>3); //3 position to right

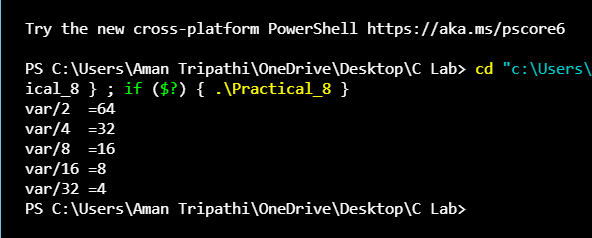
printf("var/16 =%d \n",var>>4); //4 position to right

printf("var/32 =%d \n",var>>5); //5 position to right

return 0;

}

Output:-



Practical 9

Code:-

#include <stdio.h>

#include <conio.h>

void main()

{

int i = 1;

printf("\nSquare of 1 to 10 Numbers :\n");

while (i <= 10)

{

printf("\n%d : %d", i, i \* i);

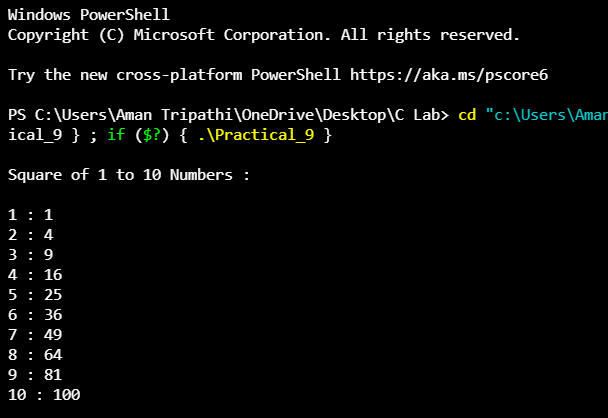
i = i + 1;

}

getch();

}

Output:-



Practical 10

Code:-

#include <stdio.h>

void main()

{

int number = 0;

printf("Enter the Number to find factor: ");

scanf("%d", &number);

printf("\n");

printf("Factors: ");

for (int i = 1; i <= number; i++)

{

if (number % i == 0)

{

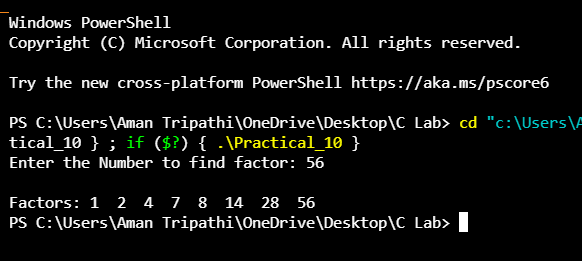
printf("%d ", i);

}

}

}

Output:-



Practical 11

Code:-

#include <stdio.h>

void main()

{

int num1 = 0;

int num2 = 0;

int num3 = 0;

printf("Enter the value of Num1 Num2 Num3 :-\n");

scanf("%d\n%d\n%d", &num1, &num2, &num3);

if (num1 >= num2 && num1 >= num3)

{

printf("Greatest Number is %d.", num1);

}

else if (num2 >= num1 && num2 >= num3)

{

printf("Greatest Number is %d.", num2);

}

else if (num3 >= num1 && num3 >= num2)

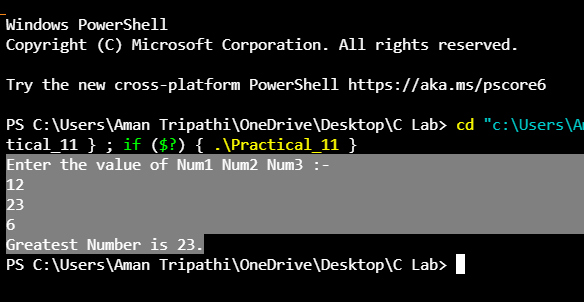
{

printf("Greatest Number is %d.", num3);

}

}

Output:-



Practical 12

Code:-

#include <stdio.h>

void main()

{

int a, b, c, big;

printf("Enter three numbers : \n");

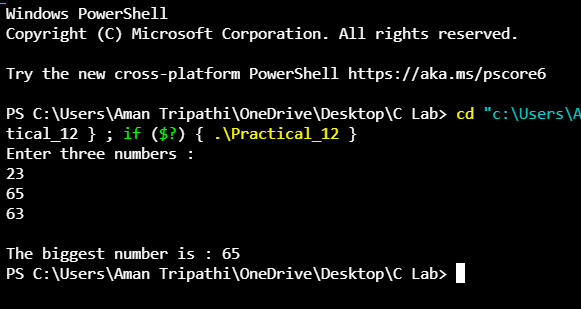
scanf("%d %d %d", &a, &b, &c);

big = a > b ? (a > c ? a : c) : (b > c ? b : c);

printf("\nThe biggest number is : %d", big);

}

Output:-



Practical 13

Code:-

#include <stdio.h>

void main()

{

int N = 0;

printf("How Many Number you want to check:- ");

scanf("%d", &N);

for (int i = 1; i <= N; i++)

{

float number;

printf("Enter the value for Number %d:- ", i);

scanf("%f", &number);

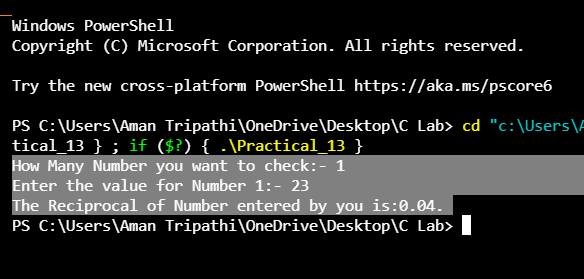
float recip = (1 / number);

printf("The Reciprocal of Number entered by you is:%0.2f.\n", recip);

}

}

Output:-



Practical 14

Code:-

// \*

// \*\*\*

// \*\*\*\*\*

// \*\*\*\*\*\*\*

//\*\*\*\*\*\*\*\*\*

#include <stdio.h>

int main()

{

int rows;

printf("Enter the Value of rows: ");

scanf("%d", &rows);

for (int i = 1; i <= rows; i++)

{

for (int space = 1; space <= (rows - i); space++)

{

printf(" ");

}

for (int star = 1; star <= (2 \* i - 1); star++)

{

printf("\*");

}

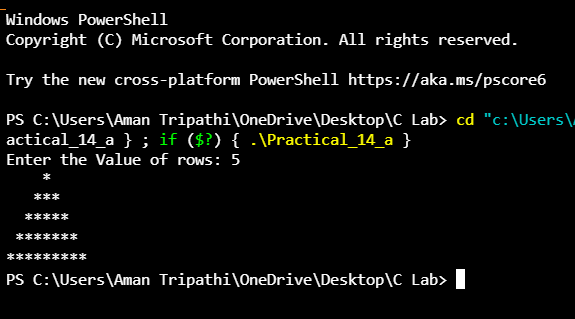
printf("\n");

}

return 0;

}

Output:-



Code:-

// \*

// \*\*

// \*\*\*

// \*\*\*\*

//\*\*\*\*\*

#include <stdio.h>

int main()

{

int rows;

printf("Enter the Value of rows: ");

scanf("%d", &rows);

for (int i = 1; i <= rows; i++)

{

int space;

for (space = 1; space <= (rows - i); space++)

{

printf(" ");

}

for (int star = 0; star <= (5 - space); star++)

{

printf("\*");

}

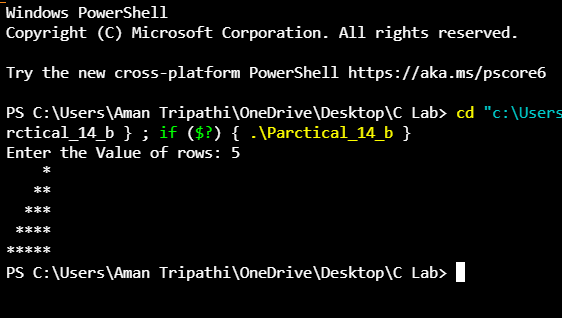
printf("\n");

}

return 0;

}

Output:-



Code:-

//\*

//\*\*

//\*\*\*

//\*\*\*\*

//\*\*\*\*\*

#include <stdio.h>

int main()

{

int rows;

printf("Enter the Value of rows: ");

scanf("%d", &rows);

for (int i = 1; i <= rows; i++)

{

for (int star = 1; star <= i; star++)

{

printf("\*");

}

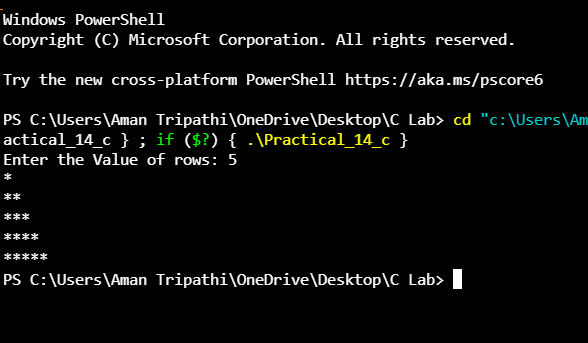
printf("\n");

}

return 0;

}

Output:-



Code:-

//\*\*\*\*\*\*\*\*\*

// \*\*\*\*\*\*\*

// \*\*\*\*\*

// \*\*\*

// \*

#include <stdio.h>

void main(){

int rows;

printf("Enter the Value of rows: ");

scanf("%d",&rows);

for (int i = rows; i >= 1; i--)

{

for(int space=1;space<=rows-i;space++){

printf(" ");

}

for (int j = 1; j <= 2\*i-1; j++)

{

printf("\*");

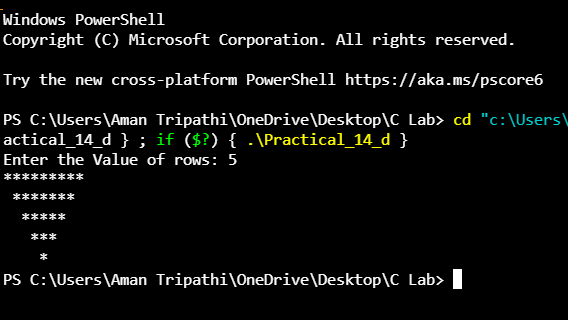
}

printf("\n");

}

}

Output:-



Code:-

// \*

// \*\*\*

// \*\*\*\*\*

// \*\*\*\*\*\*\*

//\*\*\*\*\*\*\*\*\*

// \*\*\*\*\*\*\*

// \*\*\*\*\*

// \*\*\*

// \*

#include <stdio.h>

int main()

{

int rows;

printf("Enter the Value of rows: ");

scanf("%d", &rows);

for (int i = 1; i <= rows; i++)

{

for (int space = 1; space <= (rows - i); space++)

{

printf(" ");

}

for (int star = 1; star <= (2 \* i - 1); star++)

{

printf("\*");

}

printf("\n");

}

for (int i = rows - 1; i >= 1; i--)

{

for (int space = 1; space <= rows - i; space++)

{

printf(" ");

}

for (int j = 1; j <= 2 \* i - 1; j++)

{

printf("\*");

}

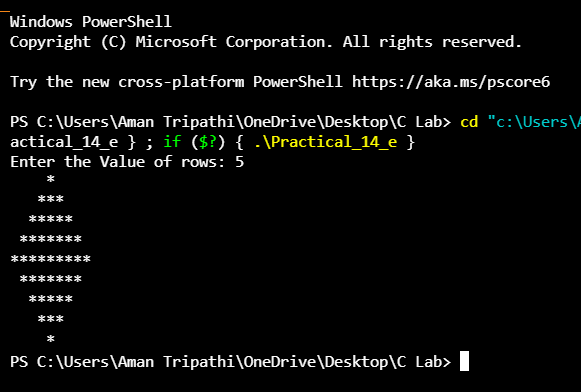
printf("\n");

}

return 0;

}

Output:-



Practical 15

Code:-

#include <stdio.h>

void main(){

int num;

int reverse= 0;

printf("Enter the number for reverse\n");

scanf("%d",&num);

while (num>0)

{

int d = num%10;

reverse = reverse\*10+d;

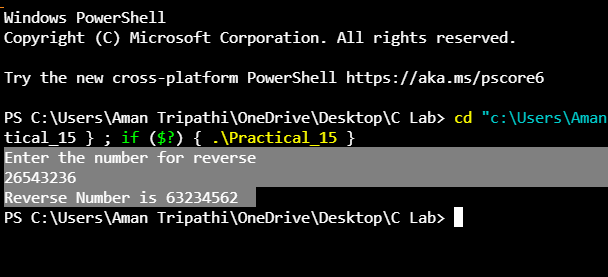
num = num/10;

}

printf("Reverse Number is %d ",reverse);

}

Output:-



Practical 16

Code:-

// Check the Entered number is character or Special char or number

#include <stdio.h>

void main()

{

char ch = ' ';

printf("Enter the Character for check:-");

scanf("%c", &ch);

if (ch >= 97 && ch <= 122)

{

printf("The Entered Character is Small Alphabet.");

}

else if (ch >= 65 && ch <= 90)

{

printf("The Entered Character is Capital Alphabet.");

}

else if (ch >= 32 && ch <= 47 || ch >= 58 && ch <= 64 || ch >= 91 && ch <= 96 || ch >= 123 && ch <= 126)

{

printf("The Entered Character is Special Character.");

}

else if (ch >= 0 && ch <= 255)

{

printf("The Entered Character is Number.");

}

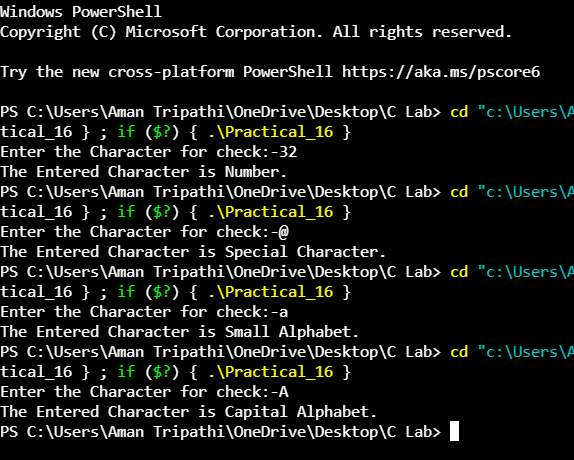
else{

printf("Wrong Input");

}

}

Output:-



Practical 17

Code:-

#include <stdio.h>

void funA();

void funB();

void funC();

void funD();

void main()

{

printf("Hey from main function\n");

funA();

}

void funA()

{

printf("Hey from function A \n");

funB();

}

void funB()

{

printf("Hey from function B \n");

funC();

}

void funC()

{

printf("Hey from function C \n");

funD();

}

void funD()

{

printf("Hey from function D \n");

}

/\*

OUTPUT:

Hey from main function

Hey from function A

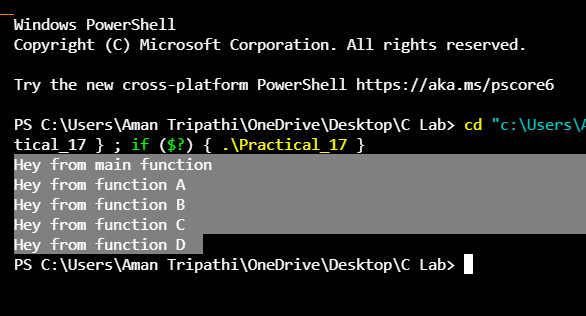
Hey from function B

Hey from function C

Hey from function D

\*/

Output:-



Practical 18

Code:-

#include <stdio.h>

#include <stdlib.h>

biggestNumber(int, int, int); // function prototype

int main()

{

int a, b, c;

printf("Enter the three numbers\n");

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

// read the numbers from user

int result = biggestNumber(a, b, c); // function call

printf("Biggest number is: %d\n", result);

// display the output on the screen

getch();

return 0;

}

int biggestNumber(int a, int b, int c)

{ // function definition with parameter

if (a > b)

{

if (a > c)

return a;

else

return c;

}

else

{

if (b > c)

return b;

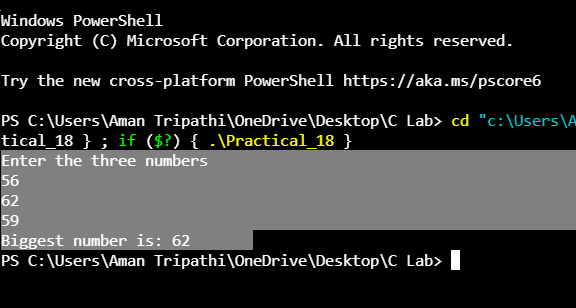
else

return c;

}

}

Output:-



Practical 19

Code:-

#include <stdio.h>

int prime(int num);

void main()

{

int number = 0;

printf("Enter the Number to check");

scanf("%d", &number);

int check = prime(number);

if (check == 0)

{

printf("Not a Prime Number");

}

else

{

printf("Prime Number");

}

}

int prime(int num)

{

int c = 0;

if (num == 0 || num == 1)

return 0;

for (int i = 1; i <= num; i++)

{

if (num % i == 0)

c++;

}

if (c == 2)

{

return 1;

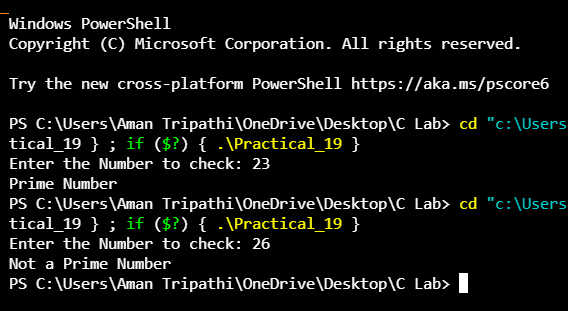
}

else

return 0;

}

Output:-



Practical 20

Code:-

#include <stdio.h>

int main()

{

int i, a = 1, count;

while (a <= 100)

{

count = 0;

i = 2;

while (i <= a / 2)

{

if (a % i == 0)

{

count++;

break;

}

i++;

}

if (count == 0 && a != 1)

{

printf(" %d ", a);

}

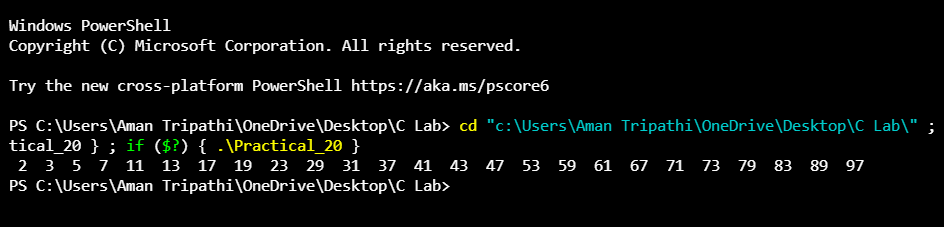
a++;

}

return 0;

}

Output:-



Practical 21

Code:-

#include <stdio.h>

int hcf(int x, int y);

void main()

{

int a, b, d;

printf("Enter 2 Numbers : ");

scanf("%d%d", &a, &b);

if (a > b)

{

d = hcf(a, b);

}

else

{

d = hcf(b, a);

}

printf("HCF : %d", d);

}

int hcf(int x, int y)

{

int r = 1;

while (r != 0)

{

r = x % y;

x = y;

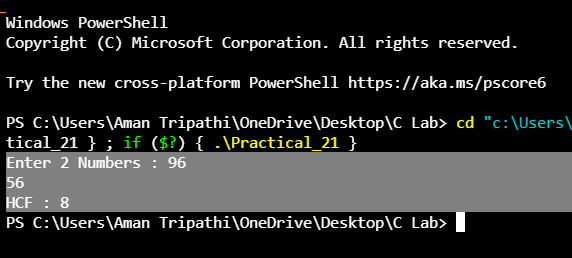
y = r;

}

return (x);

}

Output:-



Practical 22

Code:-

// Multiplication of a matrix

#include <stdio.h>

void enterData(int firstMatrix[][10], int secondMatrix[][10], int rowFirst, int columnFirst, int rowSecond, int columnSecond);

void multiplyMatrices(int firstMatrix[][10], int secondMatrix[][10], int multResult[][10], int rowFirst, int columnFirst, int rowSecond, int columnSecond);

void display(int mult[][10], int rowFirst, int columnSecond);

int main()

{

int firstMatrix[10][10], secondMatrix[10][10], mult[10][10], rowFirst, columnFirst, rowSecond, columnSecond, i, j, k;

printf("Enter rows and column for first matrix: ");

scanf("%d %d", &rowFirst, &columnFirst);

printf("Enter rows and column for second matrix: ");

scanf("%d %d", &rowSecond, &columnSecond);

// If colum of first matrix in not equal to row of second matrix, asking user to enter the size of matrix again.

while (columnFirst != rowSecond)

{

printf("Error! column of first matrix not equal to row of second.\n");

printf("Enter rows and column for first matrix: ");

scanf("%d%d", &rowFirst, &columnFirst);

printf("Enter rows and column for second matrix: ");

scanf("%d%d", &rowSecond, &columnSecond);

}

// Function to take matrices data

enterData(firstMatrix, secondMatrix, rowFirst, columnFirst, rowSecond, columnSecond);

// Function to multiply two matrices.

multiplyMatrices(firstMatrix, secondMatrix, mult, rowFirst, columnFirst, rowSecond, columnSecond);

// Function to display resultant matrix after multiplication.

display(mult, rowFirst, columnSecond);

return 0;

}

void enterData(int firstMatrix[][10], int secondMatrix[][10], int rowFirst, int columnFirst, int rowSecond, int columnSecond)

{

int i, j;

printf("\nEnter elements of matrix 1:\n");

for (i = 0; i < rowFirst; ++i)

{

for (j = 0; j < columnFirst; ++j)

{

printf("Enter elements a%d%d: ", i + 1, j + 1);

scanf("%d", &firstMatrix[i][j]);

}

}

printf("\nEnter elements of matrix 2:\n");

for (i = 0; i < rowSecond; ++i)

{

for (j = 0; j < columnSecond; ++j)

{

printf("Enter elements b%d%d: ", i + 1, j + 1);

scanf("%d", &secondMatrix[i][j]);

}

}

}

void multiplyMatrices(int firstMatrix[][10], int secondMatrix[][10], int mult[][10], int rowFirst, int columnFirst, int rowSecond, int columnSecond)

{

int i, j, k;

// Initializing elements of matrix mult to 0.

for (i = 0; i < rowFirst; ++i)

{

for (j = 0; j < columnSecond; ++j)

{

mult[i][j] = 0;

}

}

// Multiplying matrix firstMatrix and secondMatrix and storing in array mult.

for (i = 0; i < rowFirst; ++i)

{

for (j = 0; j < columnSecond; ++j)

{

for (k = 0; k < columnFirst; ++k)

{

mult[i][j] += firstMatrix[i][k] \* secondMatrix[k][j];

}

}

}

}

void display(int mult[][10], int rowFirst, int columnSecond)

{

int i, j;

printf("\nOutput Matrix:\n");

for (i = 0; i < rowFirst; ++i)

{

for (j = 0; j < columnSecond; ++j)

{

printf("%d ", mult[i][j]);

if (j == columnSecond - 1)

printf("\n\n");

}

}

}

Output:-



Practical 23

Code:-

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

#define ARRAY\_SIZE 15

int leng(char \*ipstr)

{

int i=0,length=0;

for(i=0;ipstr[i]!='\0';i++)

{

length++;

}

return length;

}

char \*concat(char \*str1,char \*str2)

{

int leng1=0,leng2=0,i=0,j=0;

leng1=leng(str1);

leng2=leng(str2);

for(i=leng1;str2[j]!='\0';i++,j++)

{

str1[i]=str2[j];

}

str1[i]='\0';

return str1;

}

int compare(char \*str1,char \*str2)

{

int leng1=0,leng2=0,i=0,count=0;

leng1=leng(str1);

leng2=leng(str2);

if(leng1==leng2)

{

for(i=0;i<leng1;i++)

{

if(str1[i]!=str2[i])

{

return -1;

}

else

{

count++;

}

}

if(count==leng1)

{

return 0;

}

}

else

{

return leng1-leng2;

}

}

char \*reverse(char \*str1)

{

int leng1=0,i=0,j=0,midleng;

char temp;

leng1=leng(str1);

j=leng1-1;

midleng=leng1/2;

for(i=0;i<midleng;i++)

{

temp=str1[i];

str1[i]=str1[j];

str1[j]=temp;

j=j-1;

}

return str1;

}

int main()

{

char str1[ARRAY\_SIZE],str2[ARRAY\_SIZE],\*resultstr;

int length=0,i=0,choice=0,result=-99;

printf("\n\n\tChoose the operation you need to perform...");

printf("\n\n\t1.StringLength\n\t2.StringReverse\n\t3.StringConcatenation\n\t4.StringCompare\n\n\tYour Choice(in numbers) : ");

scanf("%d",&choice);

switch(choice)

{

case 1:

// clrscr();

printf("\n\n\tEnter the input string\n\n\t");

scanf("%s",&str1);

length=leng(str1);

printf("\n\n\t The length of the string is %d",length);

break;

case 2:

// clrscr();

printf("\n\n\tEnter the input string\n\n\t");

scanf("%s",&str1);

// char \*newstr=(char\*) malloc(10\*sizeof(char));

resultstr=reverse(str1);

printf("\n\n\t The reversed string is %s",resultstr);

break;

case 3:

//clrscr();

printf("\n\n\tEnter two input strings\n\n\t");

scanf("%s %s",&str1,&str2);

resultstr=concat(str1,str2);

printf("\n\n\t The concatenated string is %s",resultstr);

break;

case 4:

//clrscr();

printf("\n\n\tEnter two input strings\n\n\t");

scanf("%s %s",&str1,&str2);

result=compare(str1,str2);

printf("\n\n\t The result is %d",result);

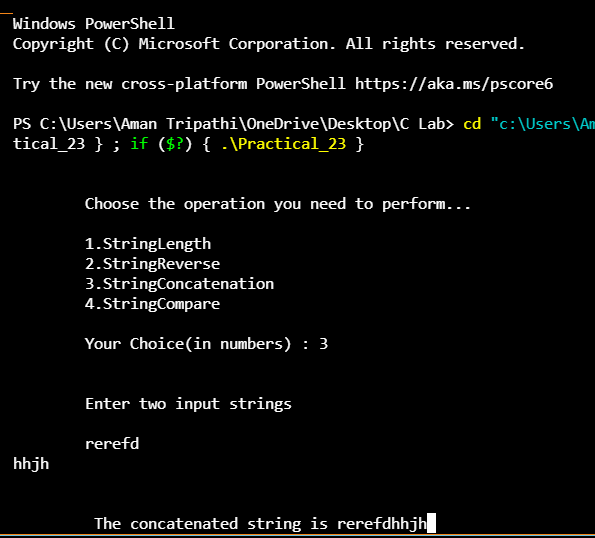
break;

}

getch();

}

Output:-



Practical 24

Code:-

#include <stdio.h>

#include <string.h>

struct bill

{

int id;

char address[200];

float amount;

} p1, p2;

int main()

{

printf("Accessing members of structure!\n\n");

p1.id = 1;

strcpy(p1.address, "Sector 41B, Market Complex, City: Siliguri, State: West Bengal");

p1.amount = 5689.36;

printf("Details of First Person!\n");

printf("Id of first person is: %d\n", p1.id);

printf("Amount due by first person is: %f\n", p1.amount);

printf("Address of first person is: %s\n", p1.address);

p2.id = 2;

strcpy(p2.address, "Sector 43B, Road No-06, Market Complex, City: Siliguri, State: West Bengal");

p2.amount = 5644.36;

printf("Details of Second Person!\n");

printf("Id of Second person is: %d\n", p2.id);

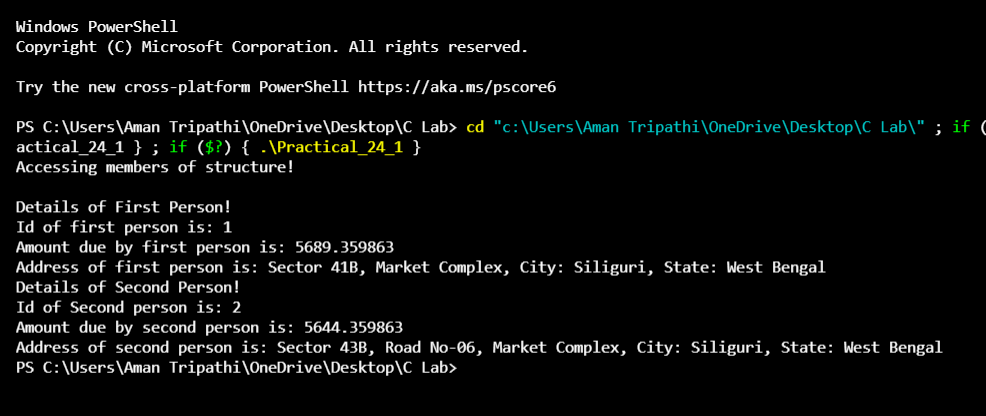
printf("Amount due by second person is: %f\n", p2.amount);

printf("Address of second person is: %s\n", p2.address);

return 0;

}

Output:-



Code:-

#include <stdio.h>

main() {

int a = 21;

int c ;

c = a;

printf("Line 1 - = Operator Example, Value of c = %d\n", c );

c += a;

printf("Line 2 - += Operator Example, Value of c = %d\n", c );

c -= a;

printf("Line 3 - -= Operator Example, Value of c = %d\n", c );

c \*= a;

printf("Line 4 - \*= Operator Example, Value of c = %d\n", c );

c /= a;

printf("Line 5 - /= Operator Example, Value of c = %d\n", c );

c = 200;

c %= a;

printf("Line 6 - %= Operator Example, Value of c = %d\n", c );

c <<= 2;

printf("Line 7 - <<= Operator Example, Value of c = %d\n", c );

c >>= 2;

printf("Line 8 - >>= Operator Example, Value of c = %d\n", c );

c &= 2;

printf("Line 9 - &= Operator Example, Value of c = %d\n", c );

c ^= 2;

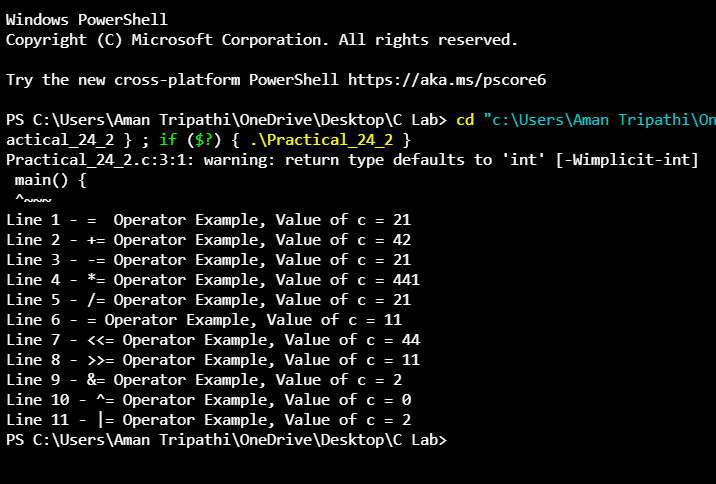
printf("Line 10 - ^= Operator Example, Value of c = %d\n", c );

c |= 2;

printf("Line 11 - |= Operator Example, Value of c = %d\n", c );

}

Output:-



Code:-

#include <stdio.h>

int main()

{

int n, i, \*ptr;

printf("\n ENTER THE NUMBER OF USN WANT TO ENTER->");

scanf("%d", &n);

ptr = (int \*)malloc(n \* sizeof(int));

if (ptr == NULL)

{

printf("\n YOUR MEMORY IS FULL");

}

printf("\n ENTER THE USN->");

for (i = 0; i < n; i++)

{

scanf("%d", ptr + i);

}

printf("\n THE USN ENTERED ARE->");

for (i = 0; i < n; i++)

{

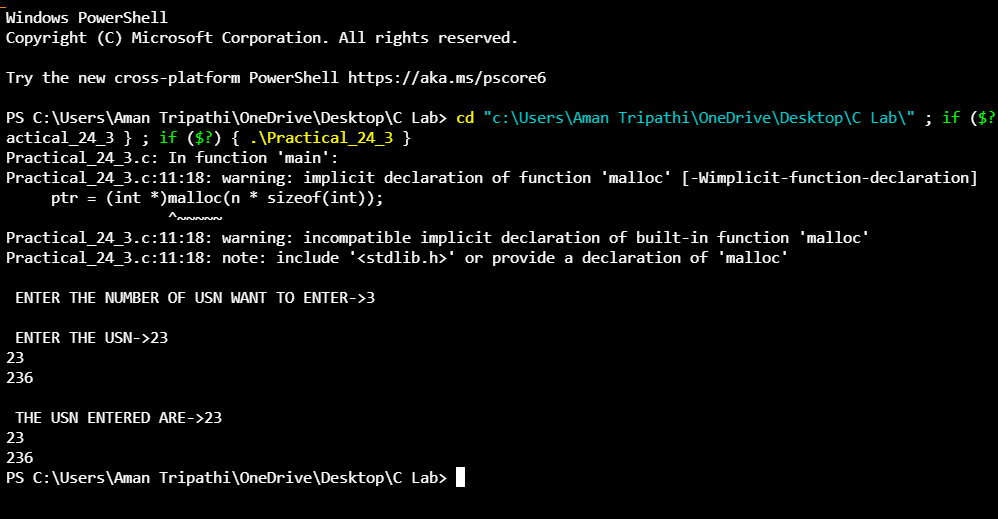
printf("%d\n", \*(ptr + i));

}

return 0;

}

Output:-



Practical 25

Code:-

#include <stdio.h>

struct student

{

char name[20];

int id;

float marks;

};

void main()

{

struct student s1, s2, s3;

int dummy;

printf("Enter the name, id, and marks of student 1 ");

scanf("%s %d %f", s1.name, &s1.id, &s1.marks);

scanf("%c", &dummy);

printf("Enter the name, id, and marks of student 2 ");

scanf("%s %d %f", s2.name, &s2.id, &s2.marks);

scanf("%c", &dummy);

printf("Enter the name, id, and marks of student 3 ");

scanf("%s %d %f", s3.name, &s3.id, &s3.marks);

scanf("%c", &dummy);

printf("Printing the details....\n");

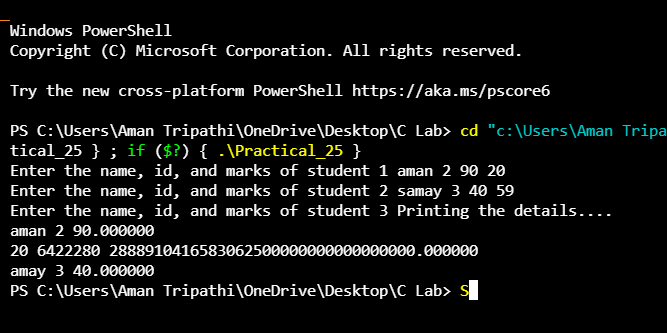
printf("%s %d %f\n", s1.name, s1.id, s1.marks);

printf("%s %d %f\n", s2.name, s2.id, s2.marks);

printf("%s %d %f\n", s3.name, s3.id, s3.marks);

}

Output:-



Practical 26

Code:-

#include <stdio.h>

#include <string.h>

// Declaration of the main

// structure

struct Organisation

{

char organisation\_name[20];

char org\_number[20];

// Declaration of the dependent

// structure

struct Employee

{

int employee\_id;

char name[20];

int salary;

// variable is created which acts

// as member to Organisation structure.

} emp;

};

// Driver code

int main()

{

struct Organisation org;

// Print the size of organisation

// structure

printf("The size of structure organisation : %ld\n",

sizeof(org));

org.emp.employee\_id = 101;

strcpy(org.emp.name, "Robert");

org.emp.salary = 400000;

strcpy(org.organisation\_name,

"GeeksforGeeks");

strcpy(org.org\_number, "GFG123768");

// Printing the details

printf("Organisation Name : %s\n",

org.organisation\_name);

printf("Organisation Number : %s\n",

org.org\_number);

printf("Employee id : %d\n",

org.emp.employee\_id);

printf("Employee name : %s\n",

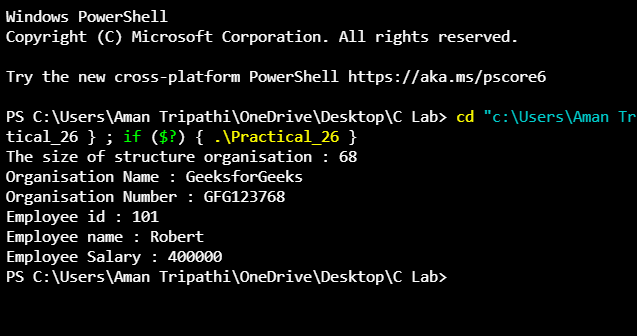
org.emp.name);

printf("Employee Salary : %d\n",

org.emp.salary);

}

Output:-



Practical 27

Code:-

#include <stdio.h>

#include <string.h>

union Data {

int i;

float f;

char str[20];

};

int main( ) {

union Data data;

data.i = 10;

data.f = 220.5;

strcpy( data.str, "C Programming");

printf( " Address - data.i : %d\n", data.i);// memory address of i

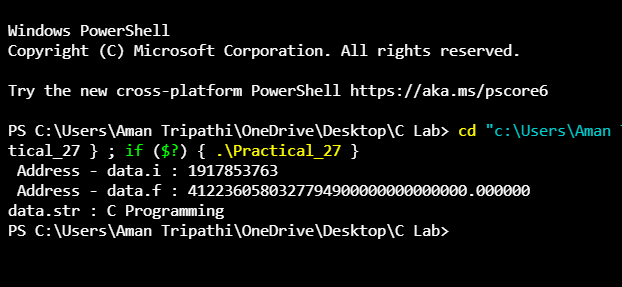
printf( " Address - data.f : %f\n", data.f);// memory address of f

printf( "data.str : %s\n", data.str);

return 0;

}

Output:-



Practical 28

Code:-

#include <stdio.h>

union abc

{

int a;

char b;

} var;

void main()

{

var.a = 65;

int \*ptr\_a = &var.a;

char \*ptr\_b = &var.b;

printf("a = %d\n", var.a);

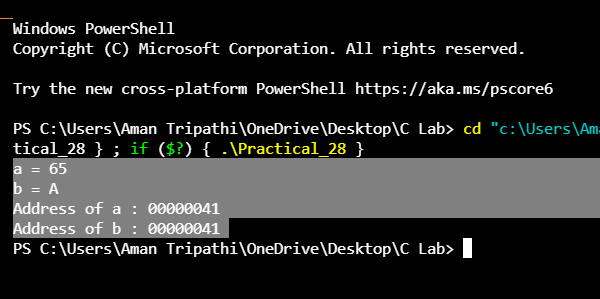
printf("b = %c\n", var.b);

printf("Address of a : %p\n",\*ptr\_a);

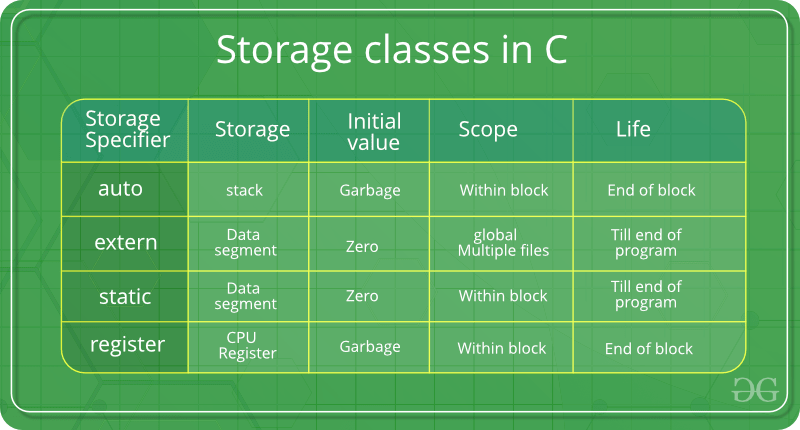
printf("Address of b : %p\n",\*ptr\_b);

}

Output:-



Practical 29



Code:-

// A C program to demonstrate different storage

// classes

#include <stdio.h>

// declaring the variable which is to be made extern

// an initial value can also be initialized to x

int x;

void autoStorageClass()

{

printf("\nDemonstrating auto class\n\n");

// declaring an auto variable (simply

// writing "int a=32;" works as well)

auto int a = 32;

// printing the auto variable 'a'

printf("Value of the variable 'a'"

" declared as auto: %d\n",

a);

printf("--------------------------------");

}

void registerStorageClass()

{

printf("\nDemonstrating register class\n\n");

// declaring a register variable

register char b = 'G';

// printing the register variable 'b'

printf("Value of the variable 'b'"

" declared as register: %d\n",

b);

printf("--------------------------------");

}

void externStorageClass()

{

printf("\nDemonstrating extern class\n\n");

// telling the compiler that the variable

// x is an extern variable and has been

// defined elsewhere (above the main

// function)

extern int x;

// printing the extern variables 'x'

printf("Value of the variable 'x'"

" declared as extern: %d\n",

x);

// value of extern variable x modified

x = 2;

// printing the modified values of

// extern variables 'x'

printf("Modified value of the variable 'x'"

" declared as extern: %d\n",

x);

printf("--------------------------------");

}

void staticStorageClass()

{

int i = 0;

printf("\nDemonstrating static class\n\n");

// using a static variable 'y'

printf("Declaring 'y' as static inside the loop.\n"

"But this declaration will occur only"

" once as 'y' is static.\n"

"If not, then every time the value of 'y' "

"will be the declared value 5"

" as in the case of variable 'p'\n");

printf("\nLoop started:\n");

for (i = 1; i < 5; i++) {

// Declaring the static variable 'y'

static int y = 5;

// Declare a non-static variable 'p'

int p = 10;

// Incrementing the value of y and p by 1

y++;

p++;

// printing value of y at each iteration

printf("\nThe value of 'y', "

"declared as static, in %d "

"iteration is %d\n",

i, y);

// printing value of p at each iteration

printf("The value of non-static variable 'p', "

"in %d iteration is %d\n",

i, p);

}

printf("\nLoop ended:\n");

printf("--------------------------------");

}

int main()

{

printf("A program to demonstrate"

" Storage Classes in C\n\n");

// To demonstrate auto Storage Class

autoStorageClass();

// To demonstrate register Storage Class

registerStorageClass();

// To demonstrate extern Storage Class

externStorageClass();

// To demonstrate static Storage Class

staticStorageClass();

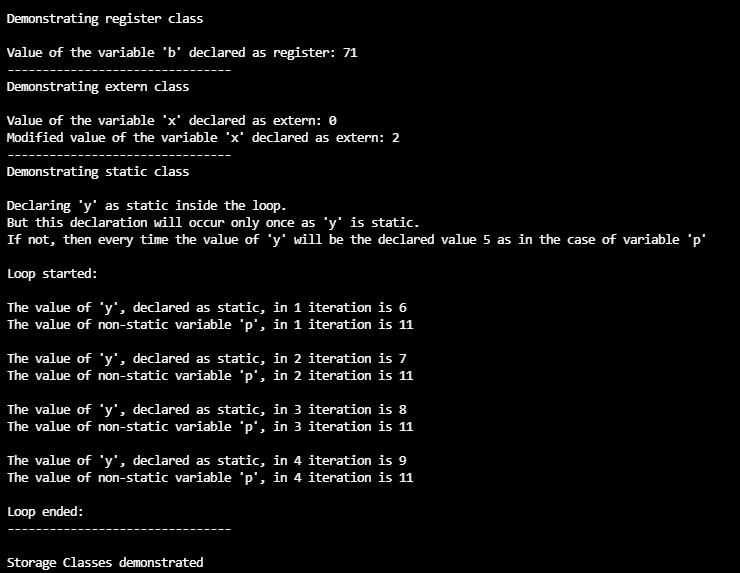
// exiting

printf("\n\nStorage Classes demonstrated");

return 0;

}

Output:-



Practical 30

Code:-

#include <stdio.h>

#include <stdlib.h>

int main()

{

int num;

FILE \*fptr;

if ((fptr = fopen("C:\\File.txt", "r")) == NULL)

{

printf("Error! opening file");

// Program exits if the file pointer returns NULL.

exit(1);

}

fscanf(fptr, "%d", &num);

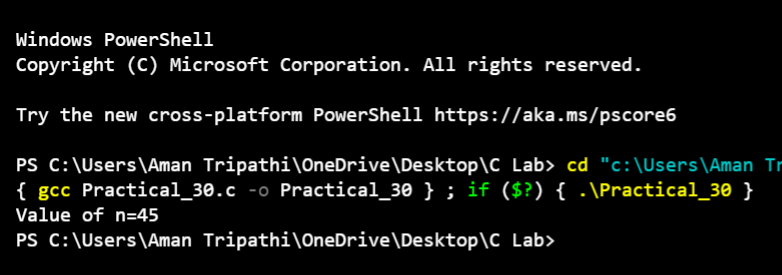
printf("Value of n=%d", num);

fclose(fptr);

return 0;

}

Output:-



Practical 31

Code:-

// C program to illustrate fgetc() function

#include <stdio.h>

int main ()

{

// open the file

FILE \*fp = fopen("File.txt","r");

// Return if could not open file

if (fp == NULL)

return 0;

do

{

// Taking input single character at a time

char c = fgetc(fp);

// Checking for end of file

if (feof(fp))

break ;

printf("%c", c);

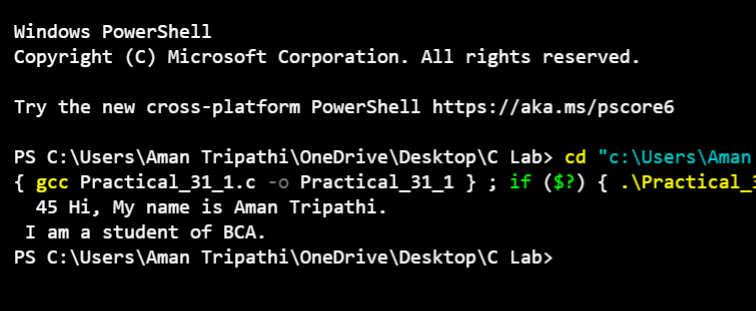
} while(1);

fclose(fp);

return(0);

}

Output:-



Code:-

// C program to illustrate fputc() function

#include<stdio.h>

int main()

{

int i = 0;

FILE \*fp = fopen("output.txt","w");

// Return if could not open file

if (fp == NULL)

return 0;

char string[] = "good bye", received\_string[20];

for (i = 0; string[i]!='\0'; i++)

// Input string into the file

// single character at a time

fputc(string[i], fp);

fclose(fp);

fp = fopen("File.txt","r");

// Reading the string from file

fgets(received\_string,20,fp);

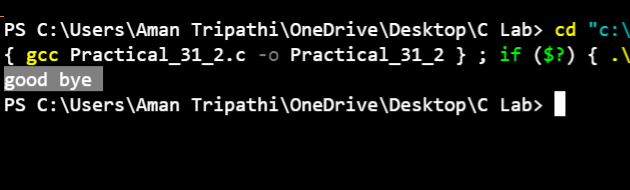
printf("%s", received\_string);

fclose(fp);

return 0;

}

Output:-



Practical 32

Code:-

#include <stdio.h>

#include <string.h>

int main()

{

FILE \*fp;

char str[80];

char c;

fp = fopen("File1.txt", "a");

FILE \*fpt;

fpt = fopen("File1.txl", "r");

printf("Enter your message:");

gets(str);

fprintf(fp, "%s", str);

printf("Your message is appended in File1.txt file.");

c = fgetc(fpt);

while (c != EOF)

{

printf("%c", c);

c = fgetc(fpt);

}

fclose(fp);

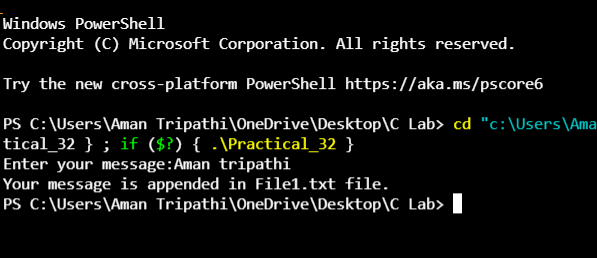
fclose(fpt);

// File validation is to be added..

return 0;

}

Output:-



Practical 33

Code:-

#include <stdio.h>

#include <stdlib.h>

int main()

{

FILE \*fp;

char str[80], str1[80];

fp = fopen("File1.txt","w");

if(fp == NULL)

{

printf("Cannot open file.\n");

exit(1);

}

printf("Enter string to be written in a file: ");

fscanf(stdin, "%s", str); /\*Read from keyboard \*/

fprintf(fp, "%s", str); /\*Write str to file \*/

fclose(fp);

fp = fopen("File1.txt","r");

if(fp == NULL) {

printf("Cannot open file.\n");

exit(1);

}

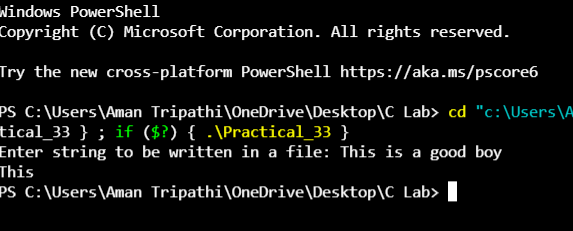
fscanf(fp, "%s", str1); /\* read a word from file and copy into str1 \*/

fprintf(stdout, "%s", str1); /\* print str1 on screen \*/

return 0;

}

Output:-



Practical 34

Code:-

#include<stdio.h>

int main(){

FILE \*fp;

char ch;

fp=fopen("File1.txt" ,"r");

fseek( fp,15,SEEK\_SET);

ch=fgetc(fp);

while(!feof(fp ))

{

printf("%c" ,ch);

printf("%d", ftell(fp ));

ch= getc(fp );

}

rewind(fp );

while(!feof(fp))

{

printf("%c" ,ch);

printf("%d" ,ftell(fp));

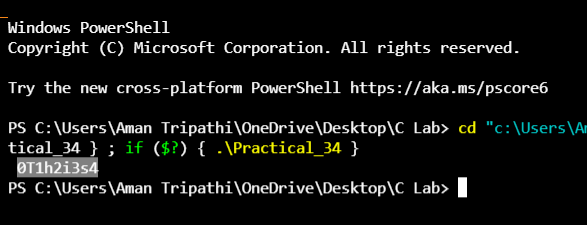
ch= fgetc(fp);

}

fclose(fp );

}

Output:-



Practical 35

Code:-

#include<stdio.h>

#include<conio.h>

void main(){

FILE \*fp;

char c;

fp=fopen("File.txt","r");

while((c=fgetc(fp))!=EOF){

printf("%c",c);

}

rewind(fp);//moves the file pointer at beginning of the file

while((c=fgetc(fp))!=EOF){

printf("%c",c);

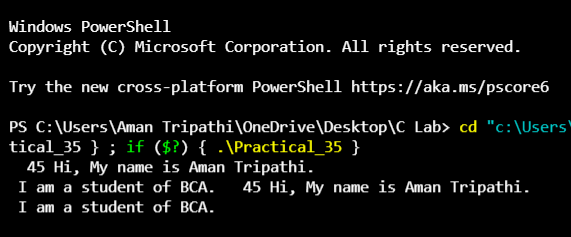
}

fclose(fp);

getch();

}

Output:-



Practical 36

Code:-

#include <stdio.h>

struct student

{

int sno;

char sname[30];

float marks;

char temp;

};

void main()

{

struct student s[60];

int i;

FILE \*fp;

fp = fopen("File2.txt", "w");

for (i = 0; i < 2; i++)

{

printf("enter details of student %d\n", i + 1);

printf("student number:");

scanf("%d", &s[i].sno);

scanf("%c", &s[i].temp);

printf("student name:");

gets(s[i].sname);

printf("student marks:");

scanf("%f", &s[i].marks);

fwrite(&s[i], sizeof(s[i]), 1, fp);

}

fclose(fp);

fp = fopen("File2.txt", "r");

for (i = 0; i < 2; i++)

{

printf("details of student %d are\n", i + 1);

fread(&s[i], sizeof(s[i]), 1, fp);

printf("student number = %d\n", s[i].sno);

printf("student name = %s\n", s[i].sname);

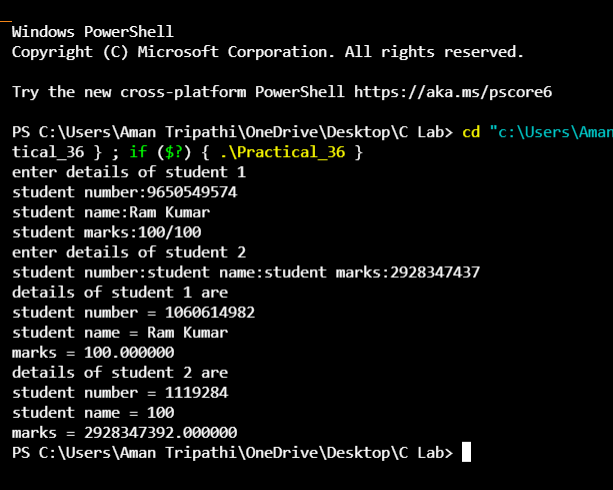
printf("marks = %f\n", s[i].marks);

}

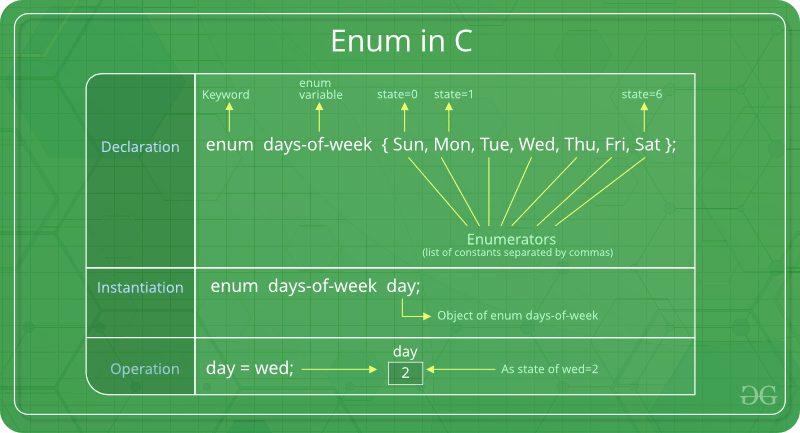
fclose(fp);

}

Output:-



Practical 37



Code:-

// An example program to demonstrate working

// of enum in C

#include<stdio.h>

enum week{Mon, Tue, Wed, Thur, Fri, Sat, Sun};

int main()

{

enum week day;

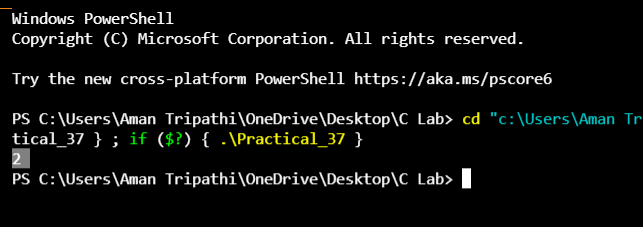
day = Wed;

printf("%d",day);

return 0;

}

Output:-



Practical 38

Code:-

#include<stdio.h>

/\*

structure is defined above all functions so it is global.

\*/

struct student

{

char name[20];

int roll\_no;

int marks;

};

void print\_struct(char name[], int roll\_no, int marks);

int main()

{

struct student stu = {"Tim", 1, 78};

print\_struct(stu.name, stu.roll\_no, stu.marks);

return 0;

}

void print\_struct(char name[], int roll\_no, int marks)

{

printf("Name: %s\n", name);

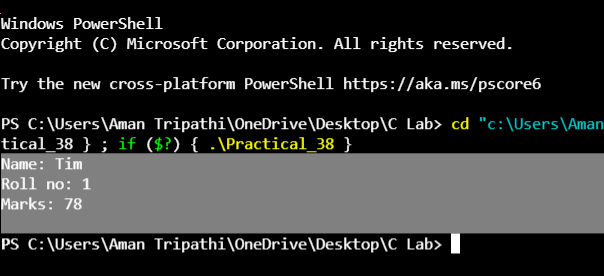
printf("Roll no: %d\n", roll\_no);

printf("Marks: %d\n", marks);

printf("\n");

}

Output:-



Code:-

#include <stdio.h>

#include <string.h>

struct student

{

int id;

char name[20];

float percentage;

};

void func(struct student \*stu1);

int main()

{

struct student stu1;

stu1.id=21;

strcpy(stu1.name, "Rambo");

stu1.percentage = 96.5;

func(&stu1);

return 0;

}

void func(struct student \*stu1)

{

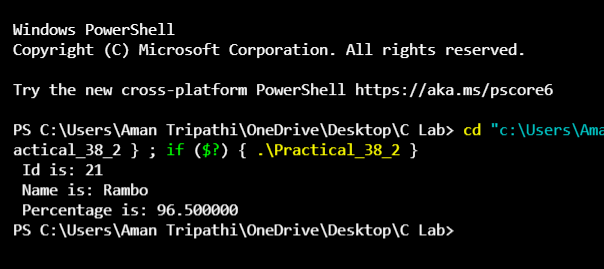
printf(" Id is: %d \n", stu1->id);

printf(" Name is: %s \n", stu1->name);

printf(" Percentage is: %f \n", stu1->percentage);

}

Output:-



Practical 39

Code:-

// C program to illustrate

// command line arguments

#include<stdio.h>

int main(int argc,char\* argv[])

{

int counter;

printf("Program Name Is: %s",argv[0]);

if(argc==1)

printf("\nNo Extra Command Line Argument Passed Other Than Program Name");

if(argc>=2)

{

printf("\nNumber Of Arguments Passed: %d",argc);

printf("\n----Following Are The Command Line Arguments Passed----");

for(counter=0;counter<argc;counter++)

printf("\nargv[%d]: %s",counter,argv[counter]);

}

return 0;

}

Output:-

