1. Write a program to check if a number is positive, negative, or zero.

IPO:

Input: number n

Process:

If n is equal to zero, n is zero, if n is greater than zero it is positive, else it is negative

Output: n is positive, negative, or zero

Code:

#include<stdio.h>  
void main()  
{

int n;  
scanf("%d",&n);  
if(n == 0)  
 printf("zero");  
else if(n > 0)  
 printf("positive");  
else  
 printf("negative");

}

Output:

1. Write a program to find the largest among three numbers.

IPO:

Input: Three numbers a, b, c

Process: if a is greater than b and greater than c, a is greater

Else if b is greater than c, b is greater else c is greater

Output: Greatest among a, b, c

Code:

#include<stdio.h>  
void main()  
{

int a,b,c;

printf(“enter three numbers”);  
scanf("%d %d %d", &a, &b, &c);  
if(a>b && a>c)  
 printf("%d is greater",a);  
else if( b>c)  
 printf("%d is greater",b);  
else  
 printf("%d is greater",c);

}

Output :



1. Write a program to check if a year is a leap year.

IPO:

Input: year say n

Process: if n % 400 is equal to zero or n % 4 equal to zero and n%100 not equal to zero; n is leap year else n is not a leap year

Output: n is leap year or not

Code:

#include<stdio.h>  
void main()  
{

int n;  
scanf("%d",&n);  
if(n % 4 == 0 && n % 100 != 0 || n % 400 == 0)  
printf("leap year");  
else  
printf("not a leap year");

}

Output:



1. Write a program to check whether a character is a vowel or consonant.

IPO:

Input: character say c

Process:

if c== 'a' or c == 'A' or c == 'e' or c == 'E' or c == 'i' or c == 'I' or c == 'o' c is a vowel

else if a is between ‘a’ and ‘z’ or ‘A’ and ‘Z’ a is consonant

else a is an invalid output

Output: a is vowel or consonant

Code:

#include<stdio.h>  
void main()  
{

char c;  
scanf("%c",&c);  
if(c== 'a' || c == 'A' || c == 'e' || c == 'E' || c == 'i' || c == 'I' || c == 'o')  
 printf("vowel");  
else if (c >= 'a' && c <= 'z' || c >= 'A' && c <= 'Z')  
 printf("consonant");  
else  
 printf("invalid input ");

}

Output:

A black background with white text

AI-generated content may be incorrect.

1. Write a program to assign grades based on marks.

IPO:

Input: mark m

Process:

If m is greater than 90 – A grade

If m is between 80 and 90 – B grade

If m is between 70 and 80 – C grade

If m is between 60 and 70 – D grade

Else – F grade

Output: grade of m

Code:

#include<stdio.h>

void main()

{

int m;

scanf("%d",&m);

if(m >= 90)

printf("A grade");

else if(m >= 80 && m < 90)

printf("B grade");

else if(m >= 70 && m< 80)

printf("C grade");

else if(m >= 60 && m < 70)

printf("D grade");

else

printf("F grade");

}

Output:

A black and white text

AI-generated content may be incorrect.

1. Write a program to check whether a number is divisible by 5 and 11.

IPO:

Input: number n

Process: if n % 5 and n % 11 is equal to zero n is divisible by both 5 and 11 else it is not divisible by 5 and 11

Output: n is divisible by both 5 and 11or not

Code:

#include<stdio.h>

void main()

{

int n;

printf("enter a number ");

scanf("%d",&n);

if(n%11 == 0 && n % 5 == 0)

printf("%d is divisible by 11 and 5",n);

else

printf("%d is not divisibleby 11 and 5",n);

}

Output:



1. Write a program to find the absolute value of a number.

IPO:

Input: number n

Process: if n is positive its absolute value is n else its absolute value is -n

Output: absolute value of n

Code:

#include<stdio.h>

void main()

{

int n;

printf(“enter a number “);

scanf(“%d”,&n);

if(n < 0)

printf(“the absolute value of %d is %d”,n,n\*-1);

else

printf(“the absolute value of %d is %d”,n,n);

}

Output:



1. Write a menu-driven program to perform +, -, \*, / operations.

IPO:

Input: two numbers say nm

Process:

If a = ‘+’, n+m

If a = ‘-‘, n-m

If a = ‘\*’, n\*m

If a = ‘/’, n/m

Output: arithmetic operation between n and m according to a

Code:

#include<stdio.h>

void main()

{

int n,m;

char a;

printf("Menu:\n1:+\n2:-\n3:\*\n4:/\nEnter your two numbers and your choice ");

scanf("%d %d %c",&n,&m,&a);

switch(a)

{

case '+': printf("%d + %d = %d",n,m,n+m); break;

case '-': printf("%d - %d = %d",n,m,n-m); break;

case '\*': printf("%d \* %d = %d",n,m,n\*m); break;

case '/': printf("%d / %d = %d",n,m,n/m); break;

}

}

Output:

A black background with white text

AI-generated content may be incorrect.

1. Write a program to find roots of a quadratic equation.

IPO:

Input: coefficients a, b, and c:

Process:

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

if d > 0

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

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

else if d == 0

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

else

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

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

Output: roots of the quadratic equation

Code:

#include <stdio.h>

#include <math.h>

int main()

{

float a, b, c;

float d, root1, root2;

float realPart, imagPart;

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

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

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

if (d > 0)

{

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

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

printf("Roots are real and distinct:\n");

printf("Root 1 = %.2f\n", root1);

printf("Root 2 = %.2f\n", root2);

}

else if (d == 0)

{

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

printf("Roots are real and equal:\n");

printf("Root 1 = Root 2 = %.2f\n", root1);

}

else

{

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

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

printf("Roots are complex and conjugate:\n");

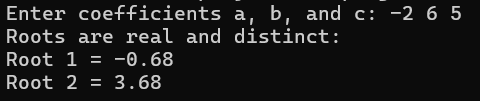
printf("Root 1 = %.2f + %.2fi\n", realPart, imagPart);

printf("Root 2 = %.2f - %.2fi\n", realPart, imagPart);

}

}

Output:



1. Write a program to find the number of digits in a number.

IPO:

Input: number n

Process:

Initializing d=0 and by using while loop and incrementing the value of d for every digit in the number

Output: number of digits in the number

Code:

#include<stdio.h>

void main()

{

int n, r, d=0;

printf("enter a number");

scanf("%d",&n);

int m = n;

while(n>0)

{

r = n % 10;

n = n / 10;

d++;

}

printf("number of digits = %d",d);

}

Output: