

Subject Name: Computer Programming with C

Subject Code: MCA102

Assignment-1

Topic: C Basics and Operators

Name: Shashwat Khaitan

Section: B

Enrollment No: 12024006015093

Class Roll No: 36

1. Write a C program to find the sum and average of three numbers.

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int a,b,c,sum;
```

```
    float avg;
```

```
    printf("enter the three numbers: ");
```

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

```
    sum= a+b+c;
```

```
    avg= sum/3.0;

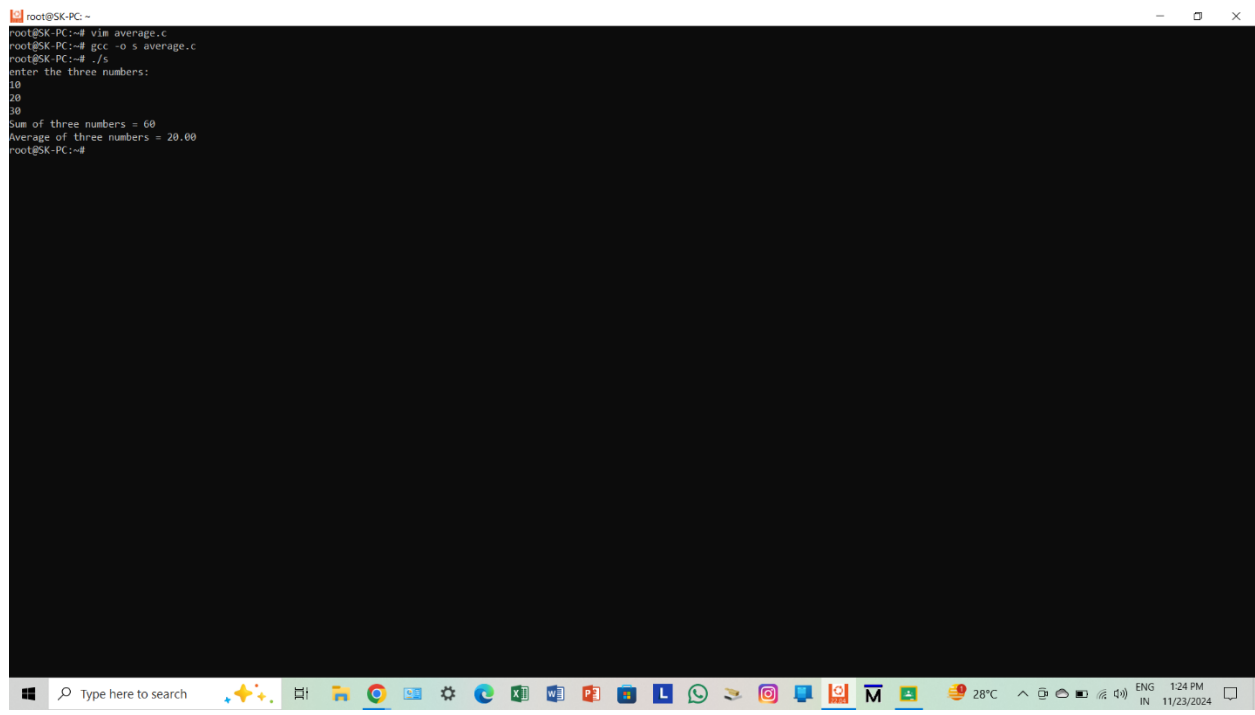
    printf("Sum of three numbers = %d\n", sum);

    printf("Average of three numbers = %.2f\n", avg);

    return 0;

}
```

OUTPUT



```
root@SK-PC: ~  
root@SK-PC:~# vim average.c  
root@SK-PC:~# gcc -o s average.c  
root@SK-PC:~# ./s  
enter the three numbers:  
10  
20  
30  
Sum of three numbers = 60  
Average of three numbers = 20.00  
root@SK-PC:~#
```

The screenshot shows a terminal window with a black background and white text. The user is in a root shell on a machine named SK-PC. They use 'vim' to edit a file named 'average.c', then 'gcc' to compile it into an executable named 's'. They run './s' which prompts for three numbers: 10, 20, and 30. The program outputs the sum (60) and the average (20.00). The Windows taskbar is visible at the bottom with various application icons and system status information.

2. Write a C program to find the sum of individual digits of a given positive integer.

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int n,sum=0,d;
```

```
    printf("enter a positive integer: ");
```

```
    scanf("%d", &n);
```

```
    while(n>0)
```

```
    {
```

```
        d=n%10;
```

```
        sum=sum+d;
```

```
        n=n/10;
```

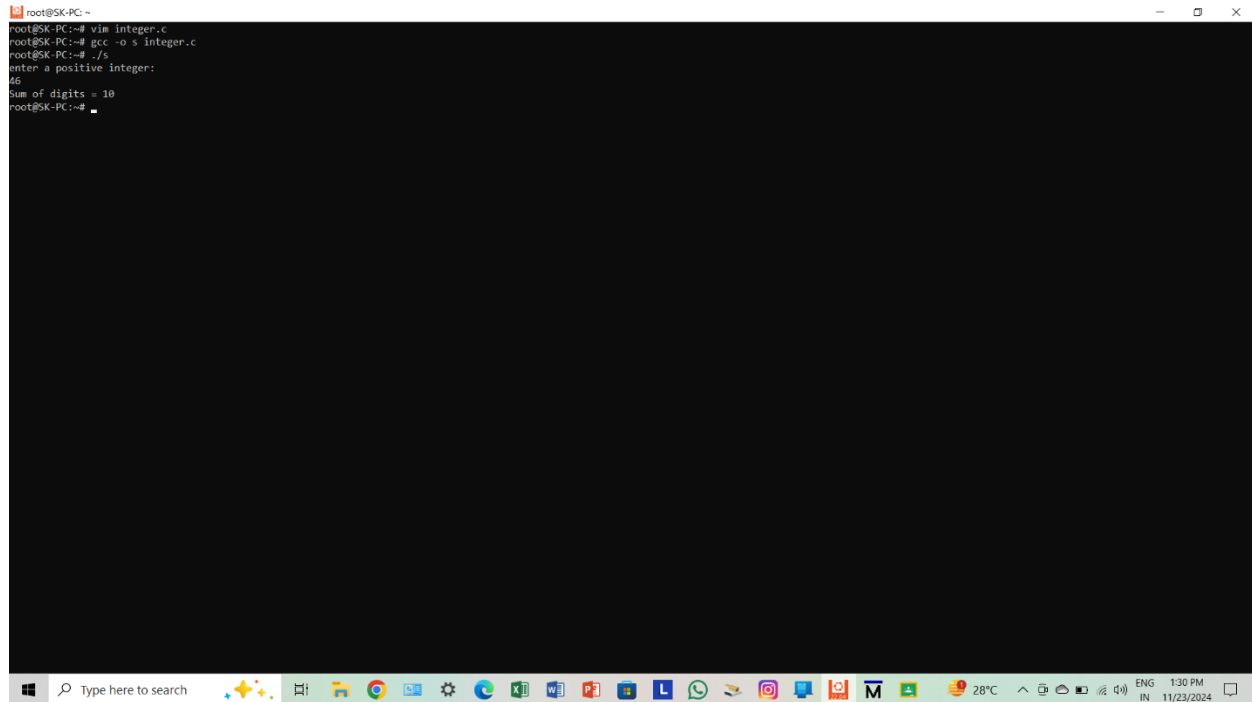
```
    }
```

```
    printf("Sum of digits = %d\n", sum);
```

```
    return 0;
```

```
}
```

OUTPUT



```
root@SK-PC: ~  
root@SK-PC:~# vim integer.c  
root@SK-PC:~# gcc -o s integer.c  
root@SK-PC:~# ./s  
enter a positive integer:  
45  
Sum of digits = 10  
root@SK-PC:~#
```

3. Write a C program to generate the first n terms of the Fibonacci sequence.

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int n,a=0,b=1,next,i;
```

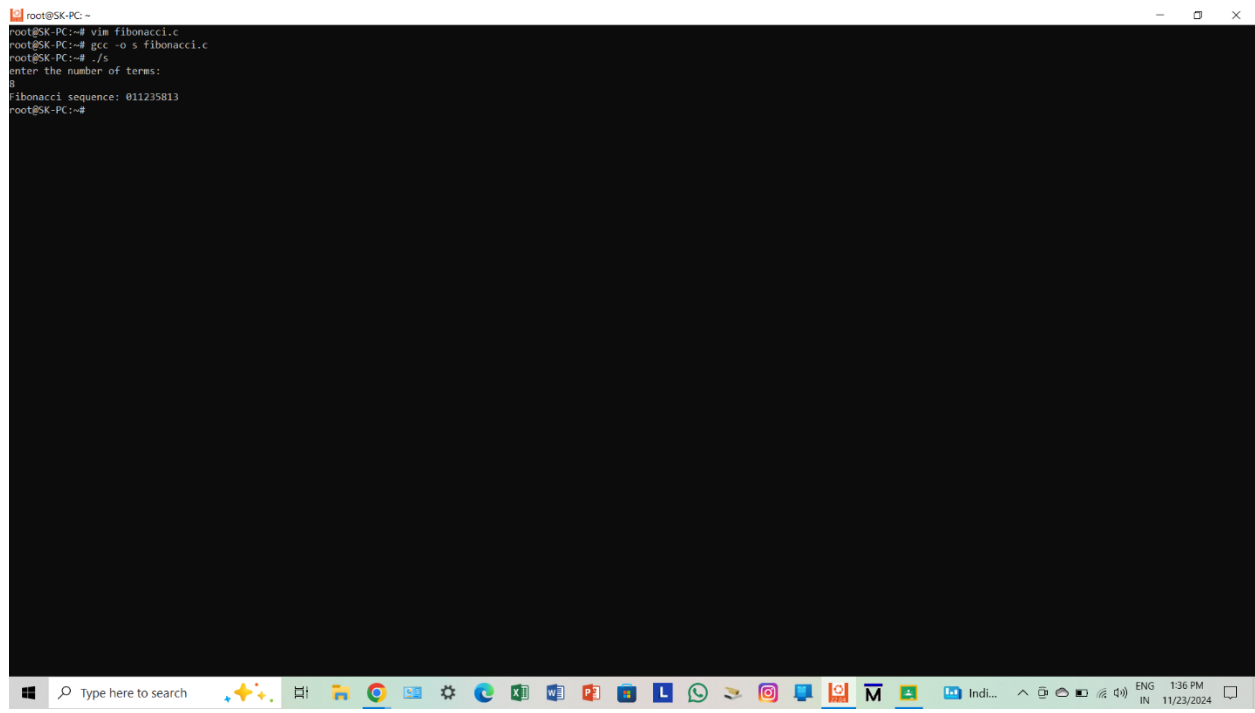
```
    printf("enter the number of terms: ");
```

```
    scanf("%d", &n);
```

```
    printf("Fibonacci sequence: ");
```

```
    for(i=1;i<=n;i++)  
    {  
        printf("%d", a);  
        next= a+b;  
        a=b;  
        b=next;  
    }  
    printf("\n");  
    return 0;  
}
```

OUTPUT



The screenshot shows a Windows terminal window with a black background and white text. The window title is "root@SK-PC: ~". The terminal output is as follows:

```
root@SK-PC:~# vim fibonacci.c  
root@SK-PC:~# gcc -o s fibonacci.c  
root@SK-PC:~# ./s  
enter the number of terms:  
8  
Fibonacci sequence: 011235813  
root@SK-PC:~#
```

The Windows taskbar is visible at the bottom of the screen, showing various application icons and the system clock indicating 1:36 PM on 11/23/2024.

4. Write a C program to generate prime numbers between 1 to n.

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int n,i,j,isPrime;
```

```
    printf("enter the value of n: ");
```

```
    scanf("%d", &n);
```

```
    printf("Prime numbers between 1 and %d: ", n);
```

```
    for(i=2;i<=n;i++)
```

```
    {
```

```
        isPrime=1;
```

```
        for(j=2;j*j<=i;j++)
```

```
        {
```

```
            if(i%j==0)
```

```
            {
```

```
                isPrime=0;
```

```
                break;
```

```
            }
```

```
        }
```

```
        if(isPrime)

            printf("%d", i);

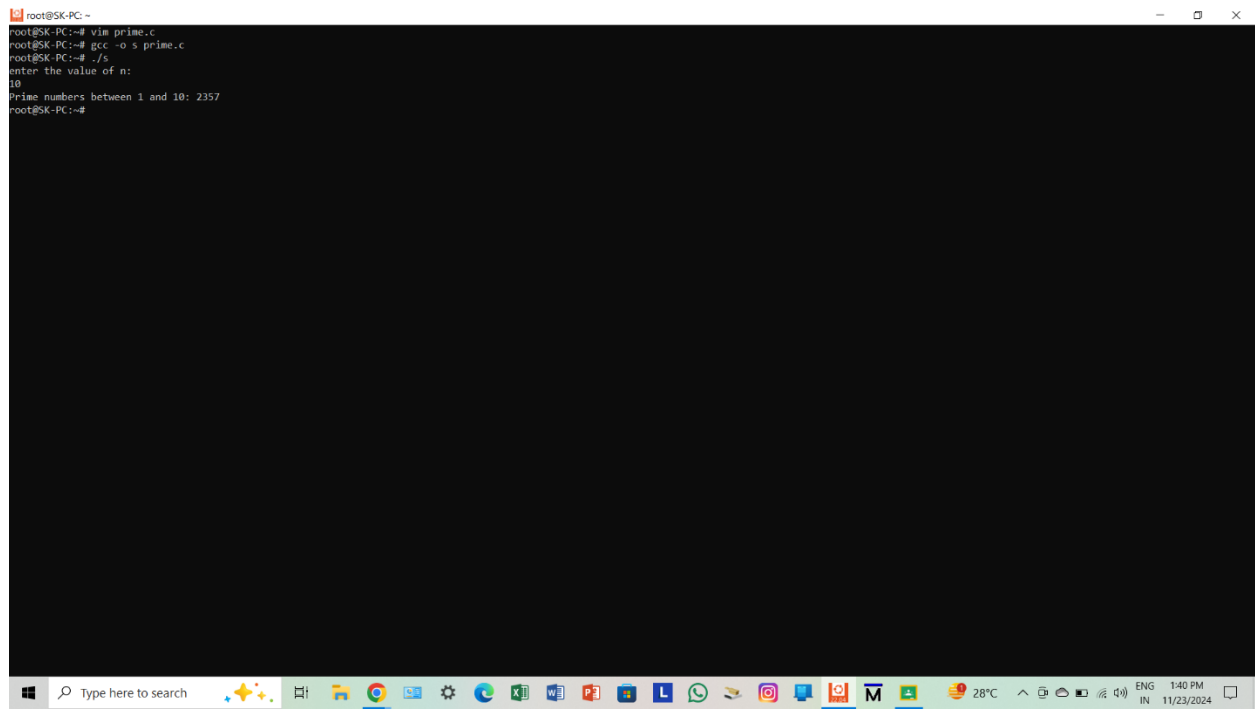
    }

    printf("\n");

    return 0;

}
```

OUTPUT



```
root@SK-PC: ~  
root@SK-PC:~# vim prime.c  
root@SK-PC:~# gcc -o s prime.c  
root@SK-PC:~# ./s  
enter the value of n:  
10  
Prime numbers between 1 and 10: 2357  
root@SK-PC:~#
```

5. Write a C program to check whether a given number is an Armstrong number or not.

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int n, original,r,sum=0;
```

```
    printf("enter a number: ");
```

```
    scanf("%d", &n);
```

```
    original=n;
```

```
    while(n>0)
```

```
    {
```

```
        r=n%10;
```

```
        sum=sum+r*r*r;
```

```
        n=n/10;
```

```
    }
```

```
    if(sum==original)
```

```
        printf("%d is an armstrong number.\n", original);
```

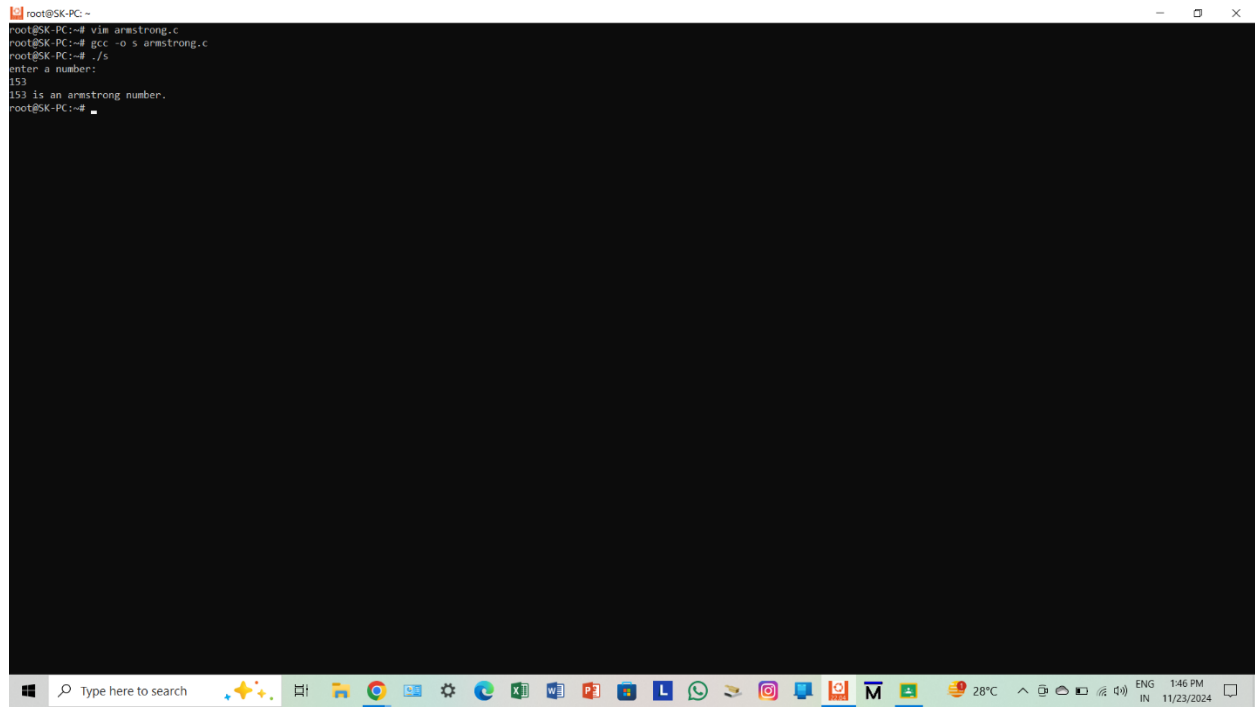
```
    else
```

```
        printf("%d is not an armstrong number.\n", original);
```



```
    return 0;
}
```

OUTPUT



```
root@SK-PC:~#  
root@SK-PC:~# vim armstrong.c  
root@SK-PC:~# gcc -o s armstrong.c  
root@SK-PC:~# ./s  
enter a number:  
153  
153 is an armstrong number.  
root@SK-PC:~#
```

The screenshot shows a terminal window with a black background and white text. The user is in a directory named SK-PC. They have created a file named armstrong.c, compiled it with gcc to produce an executable named s, and then run it. The program prompts the user to enter a number, and the user has entered 153. The program outputs that 153 is an armstrong number. The Windows taskbar is visible at the bottom of the screen.

6. Write a C program to evaluate the algebraic expression $(ax+b)/(ax-b)$.

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int a,x,b;
```

```
    float r;
```

```
printf("enter the values of a, x, and b: ");  
  
scanf("%d %d %d", &a, &x, &b);  
  
if(a*x-b!=0)  
{  
  
    r= (float)(a*x+b)/(a*x-b);  
  
    printf("Result= %.2f\n", r);  
  
}  
  
else  
  
{  
  
    printf("Division by zero not allowed.\n");  
  
}  
  
return 0;  
  
}
```

OUTPUT

```
root@SK-PC: ~  
root@SK-PC:~# vim expression.c  
root@SK-PC:~# gcc -o s expression.c  
root@SK-PC:~# ./s  
enter the values of a, x, and b:  
5  
2  
4  
Result= 2.33  
root@SK-PC:~#
```

7. Write a C program to check if the given number is a perfect number.

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int n,i,sum=0;
```

```
    printf("enter a number: ");
```

```
    scanf("%d", &n);
```

```
    for(i=1;i<n;i++)
```

```
    {
```

```
        if(n%i==0)
```

```
        sum=sum+i;

    }

    if(sum==n)

        printf("%d is a perfect number.\n", n);

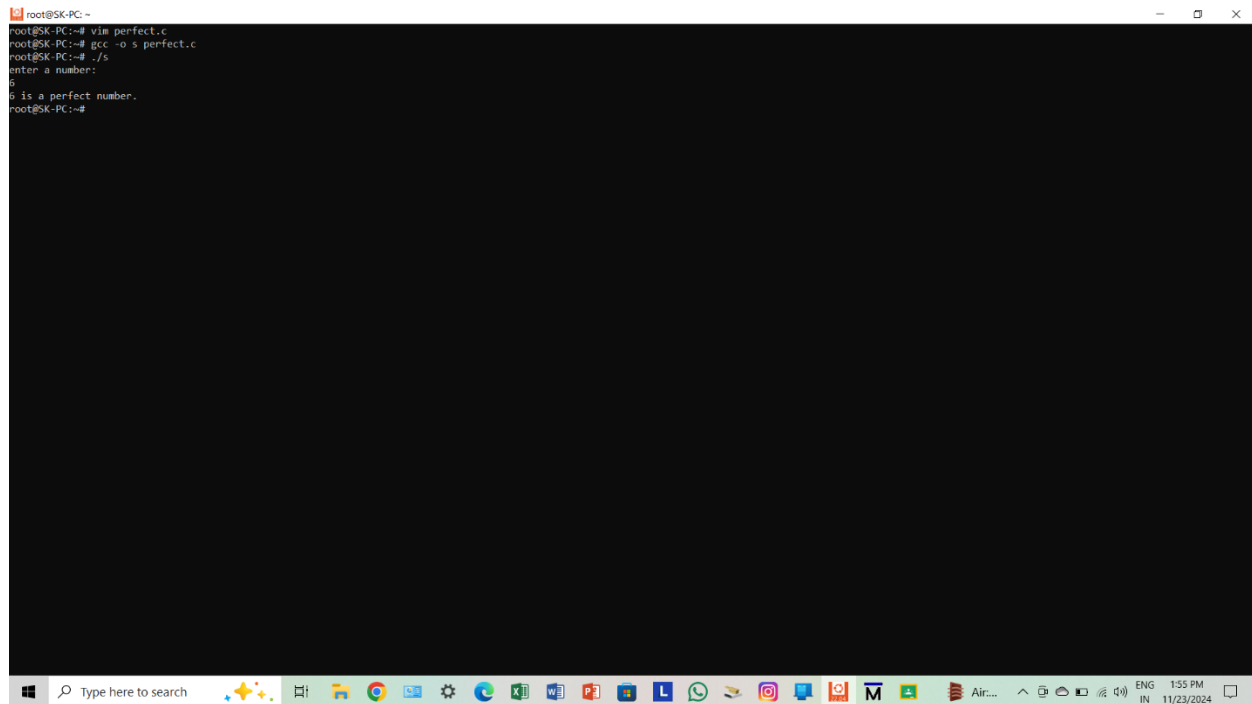
    else

        printf("%d is not a perfect number.\n", n);

    return 0;

}
```

OUTPUT



```
root@SK-PC: ~  
root@SK-PC:~# vi perfect.c  
root@SK-PC:~# gcc -o s perfect.c  
root@SK-PC:~# ./s  
enter a number:  
6  
6 is a perfect number.  
root@SK-PC:~#
```

The screenshot shows a terminal window with a black background and white text. The user is in a directory named '~' on a machine named 'SK-PC'. They have edited a file named 'perfect.c' using 'vi'. Then, they compiled the file using 'gcc -o s perfect.c' and executed the resulting program './s'. The program prompts 'enter a number:' and the user enters '6'. The program outputs '6 is a perfect number.' The terminal window is titled 'root@SK-PC: ~' and has standard window controls (minimize, maximize, close) in the top right corner. The Windows taskbar is visible at the bottom of the screen, showing various application icons and the system clock indicating 1:55 PM on 11/23/2024.

8. Write a C program to check if a given number is a strong number.

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int n, i, original, d, fact, sum=0;
```

```
    printf("enter a number: ");
```

```
    scanf("%d", &n);
```

```
    original=n;
```

```
    while(n>0)
```

```
    {
```

```
        d=n%10;
```

```
        fact=1;
```

```
        for(i=1;i<=d;i++)
```

```
        {
```

```
            fact=fact*i;
```

```
        }
```

```
        sum=sum+fact;
```

```
        n=n/10;
```

```
    }
```

```
if(sum==original)

    printf("%d is a strong number.\n", original);

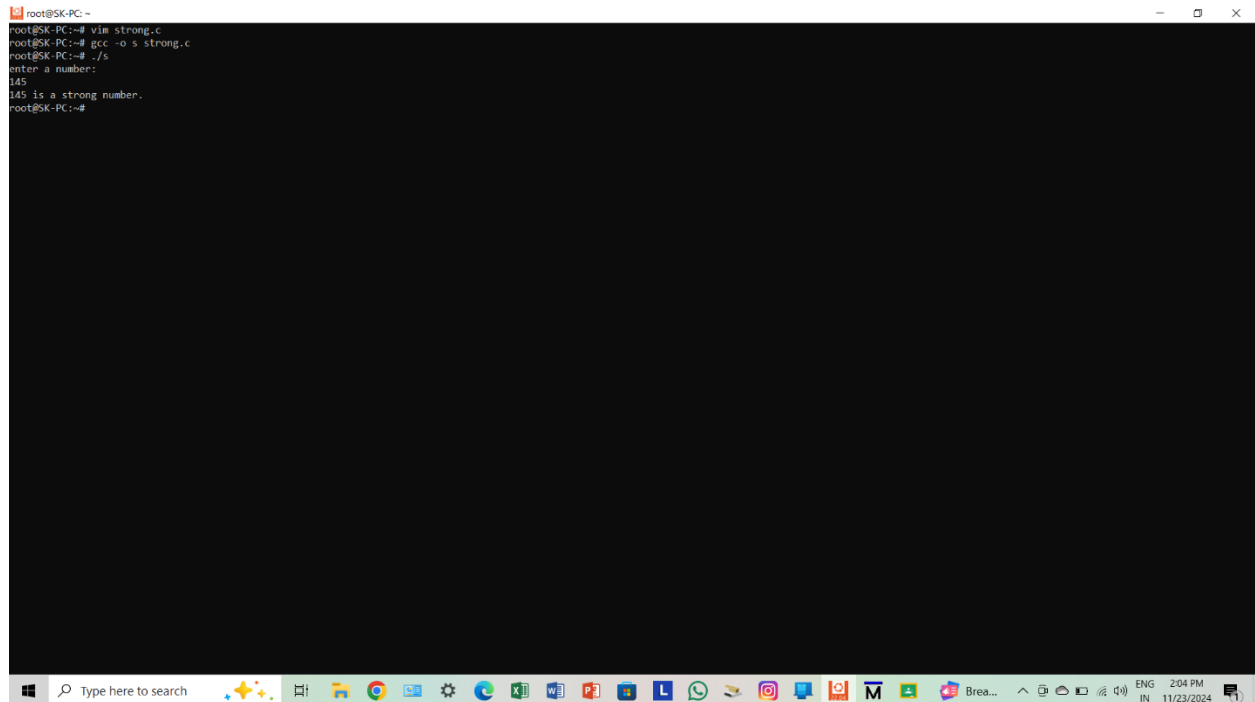
else

    printf("%d is not a strong number.\n", original);

return 0;

}
```

OUTPUT



```
root@SK-PC: ~  
root@SK-PC:~# vim strong.c  
root@SK-PC:~# gcc -o s strong.c  
root@SK-PC:~# ./s  
enter a number:  
145  
145 is a strong number.  
root@SK-PC:~#
```

9. Write a program to print your name without using any semicolons in the program.

```
#include <stdio.h>
```

```
int main()
```

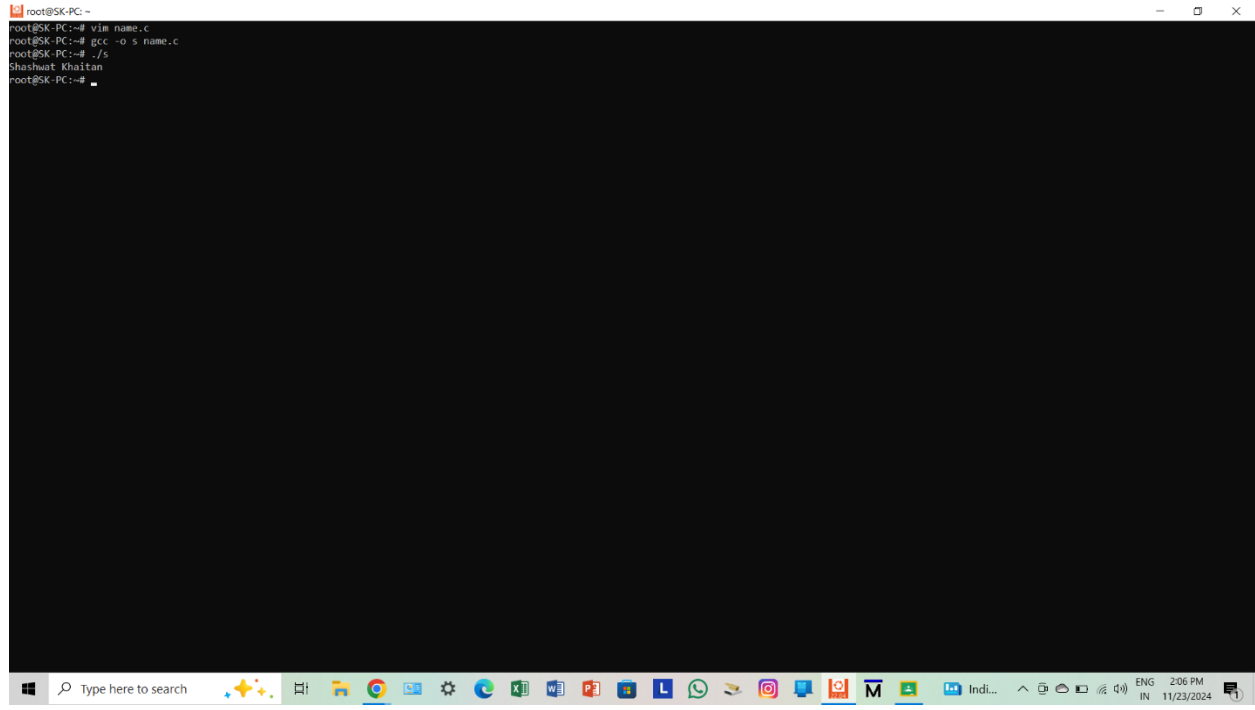
```
{
```

```
    if(printf("Shashwat Khaitan\n")){}
```

```
    return 0;
```

```
}
```

OUTPUT



```
root@SK-PC: ~  
root@SK-PC:~# vim name.c  
root@SK-PC:~# gcc -o s name.c  
root@SK-PC:~# ./s  
Shashwat Khaitan  
root@SK-PC:~#
```

The screenshot shows a terminal window with a black background and white text. The user is in a directory named '~' on a machine named 'root@SK-PC'. They have created a file 'name.c' using 'vim', compiled it with 'gcc -o s name.c', and executed it with './s'. The output of the program is 'Shashwat Khaitan'. The Windows taskbar is visible at the bottom, showing various application icons and the system clock indicating 2:06 PM on 11/23/2024.

10. Write a program to convert temperatures in Celsius to Fahrenheit and vice-versa.

```
#include <stdio.h>

int main()
{
    float c,f;

    printf("enter temperature in celsius:");

    scanf("%f", &c);

    f= (c* 9/5) + 32;

    printf("%.2f degree celsius is equal to %.2f degree fahrenheit. \n", c,f);

    return 0;
}
```

OUTPUT


```
root@SK-PC: ~  
root@SK-PC:~# vim convert.c  
root@SK-PC:~# gcc -o s convert.c  
root@SK-PC:~# ./s  
enter temperature in celsius:  
40  
40.00 degree celsius is equal to 104.00 degree fahrenheit.  
root@SK-PC:~#
```

11. Write a C program to check whether a number is a palindrome or not.

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int n, rev=0, original, d;
```

```
    printf("enter a number: ");
```

```
    scanf("%d", &n);
```

```
    original=n;
```

```
    while(n>0)
```

```
    {
```

```
    d=n%10;

    rev=rev*10+d;

    n=n/10;

}

if(rev==original)

    printf("%d is a palindrome number.\n", original);

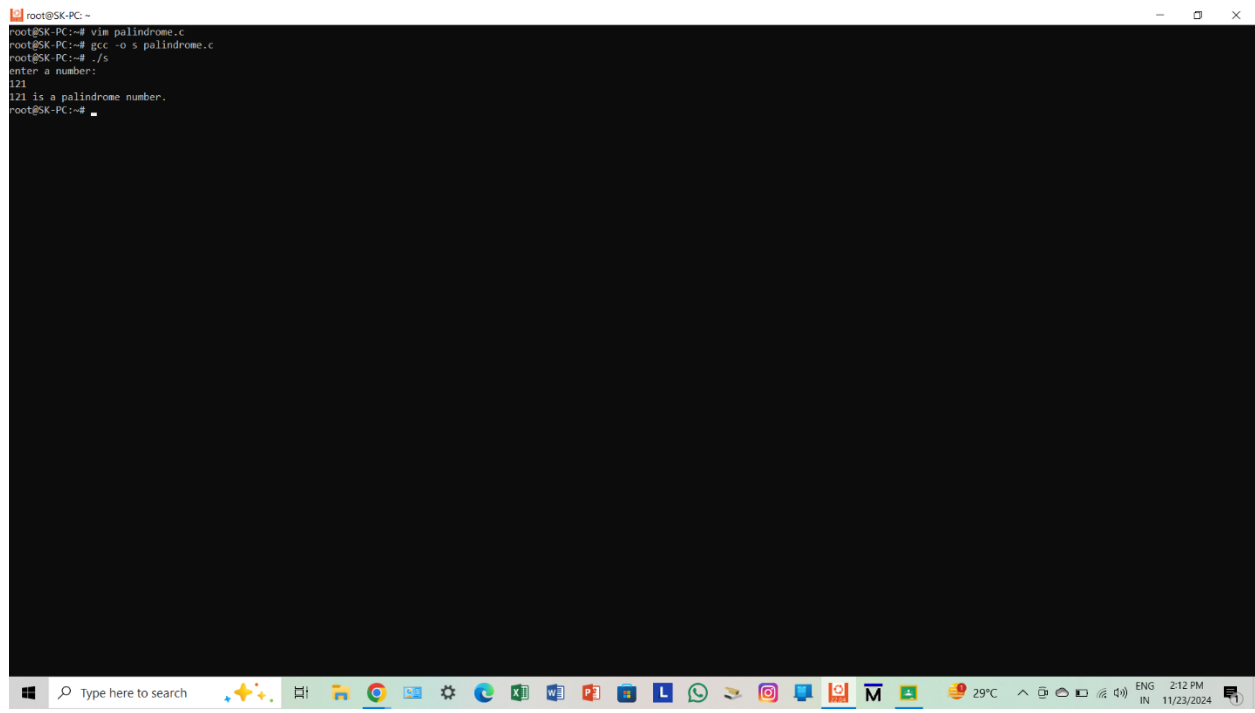
else

    printf("%d is not a palindrome number.\n", original);

return 0;

}
```

OUTPUT



```
root@SK-PC: ~  
root@SK-PC:~# vim palindrome.c  
root@SK-PC:~# gcc -o s palindrome.c  
root@SK-PC:~# ./s  
enter a number:  
121  
121 is a palindrome number.  
root@SK-PC:~#
```

12. Write a C program to find the maximum between two numbers.

```
#include <stdio.h>

int main()
{
    int a,b;

    printf("enter the two numbers: ");

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

    if(a>b)

        printf("Maximum number = %d\n", a);

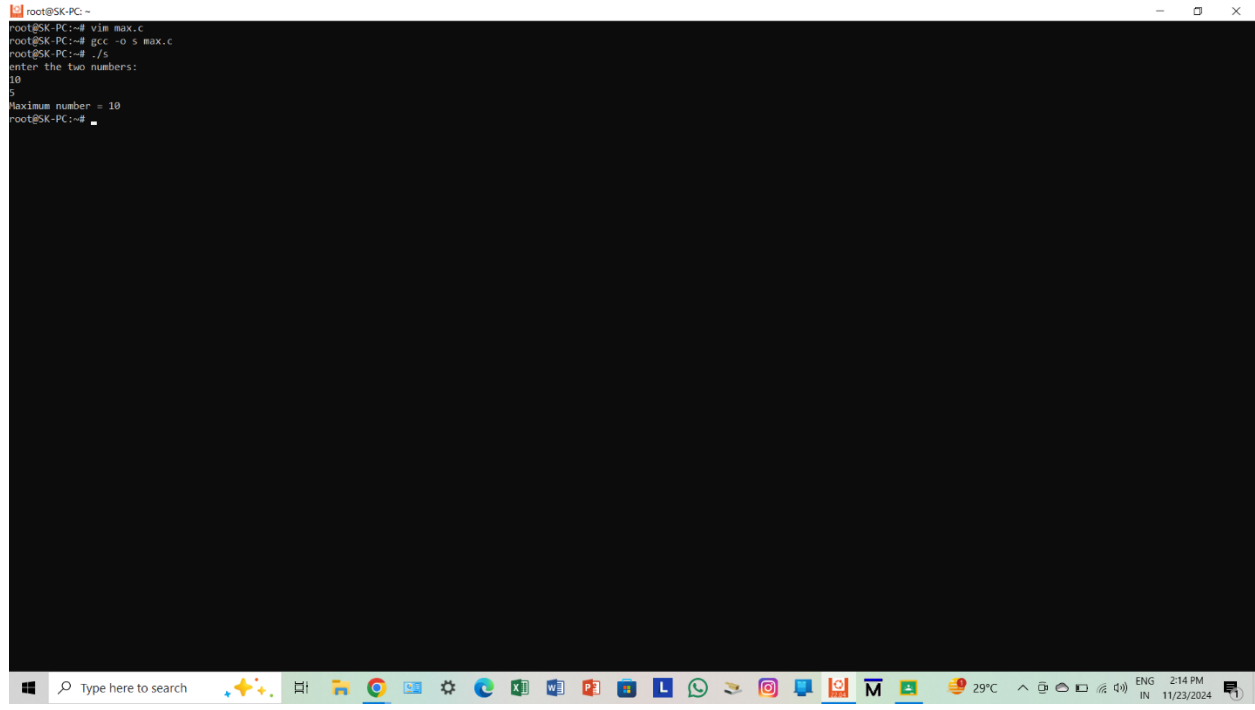
    else

        printf("Maximum number = %d\n", b);

    return 0;
}
```

OUTPUT

```
root@SK-PC: ~  
root@SK-PC:~# vim max.c  
root@SK-PC:~# gcc -o 5 max.c  
root@SK-PC:~# ./5  
enter the two numbers:  
10  
5  
Maximum number = 10  
root@SK-PC:~#
```

A screenshot of a Windows desktop environment. A terminal window is open, showing a C program being compiled and executed. The program prompts for two numbers, 10 and 5, and outputs the maximum value, 10. The Windows taskbar is visible at the bottom, featuring the Start button, a search bar, and various application icons including File Explorer, Chrome, and several social media apps. The system tray on the right shows the temperature as 29°C, the time as 2:14 PM, and the date as 11/23/2024.

13. Write a C program to find the maximum between three numbers.

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int a,b,c,max;
```

```
    printf("enter the three numbers: ");
```

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

```
    max=a;
```

```
    if(b>max)
```

```
        max=b;
```

```
    if(c>max)

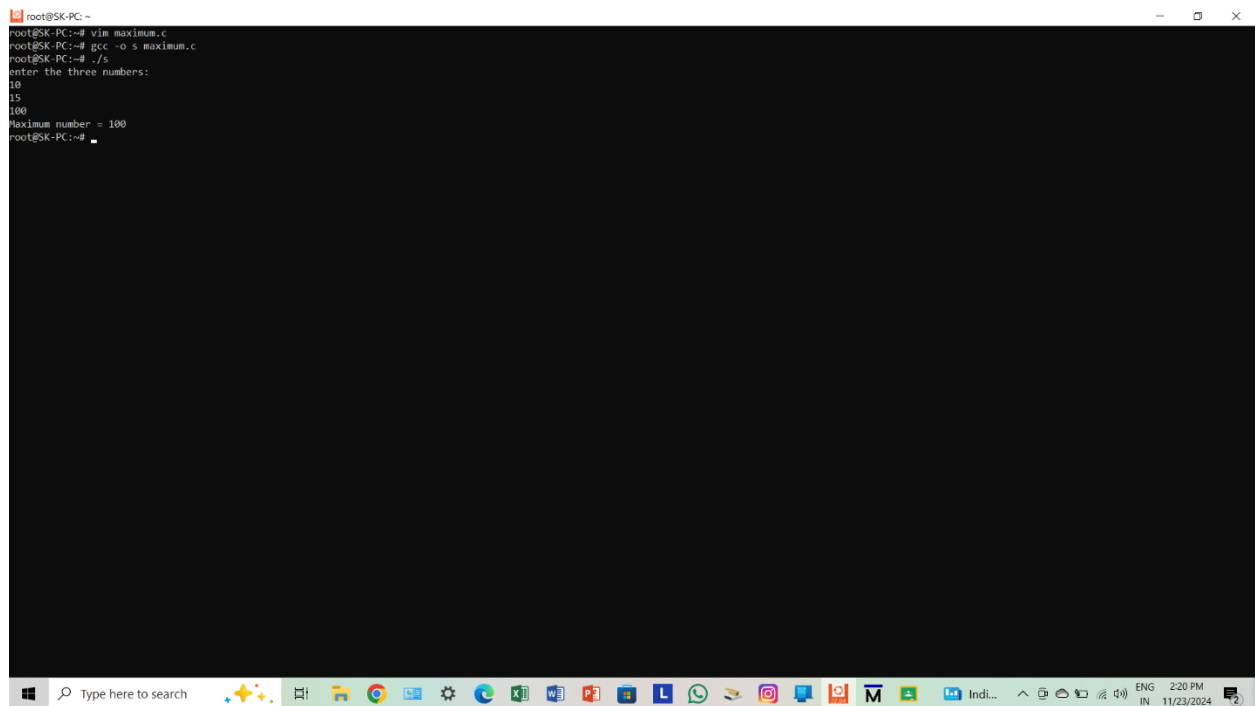
        max=c;

    printf("Maximum number = %d\n", max);

    return 0;

}
```

OUTPUT



```
root@SK-PC: ~  
root@SK-PC:~# vim maximum.c  
root@SK-PC:~# gcc -o s maximum.c  
root@SK-PC:~# ./s  
enter the three numbers:  
10  
15  
100  
Maximum number = 100  
root@SK-PC:~#
```

The screenshot shows a terminal window with a black background and white text. The user is in a root shell on a machine named SK-PC. They have edited a file named maximum.c using vim, compiled it with gcc to create an executable named s, and then run it. The program prompts the user to enter three numbers: 10, 15, and 100. It then outputs "Maximum number = 100". The terminal window is titled "root@SK-PC: ~". The Windows taskbar is visible at the bottom of the screen, showing various application icons and the system clock indicating 2:20 PM on 11/23/2024.

14. Write a C program to check whether a number is negative, positive, or zero.

```
#include <stdio.h>

int main()
{
    int n;

    printf("enter a number: ");

    scanf("%d", &n);

    if(n>0)

        printf("%d is a positive number.\n", n);

    else if(n<0)

        printf("%d is a negative number.\n", n);

    else

        printf("The number is zero.\n");

    return 0;
}
```

OUTPUT

```
root@SK-PC: ~  
root@SK-PC:~# vim pnz.c  
root@SK-PC:~# gcc -o s pnz.c  
root@SK-PC:~# ./s  
enter a number:  
8  
8 is a positive number.  
root@SK-PC:~# gcc -o s pnz.c  
root@SK-PC:~# ./s  
enter a number: -5  
-5 is a negative number.  
root@SK-PC:~#
```

15. Write a C program to check whether a number is divisible by 5 and 11 or not within the range of 100 to 500.

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int n;
```

```
    printf("enter a number (100-500): ");
```

```
    scanf("%d", &n);
```

```
    if(n>=100 && n<=500)
```

```
    {
```

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

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

    else

        printf("%d is not divisible by 5 and 11.\n", n);

}

else

{

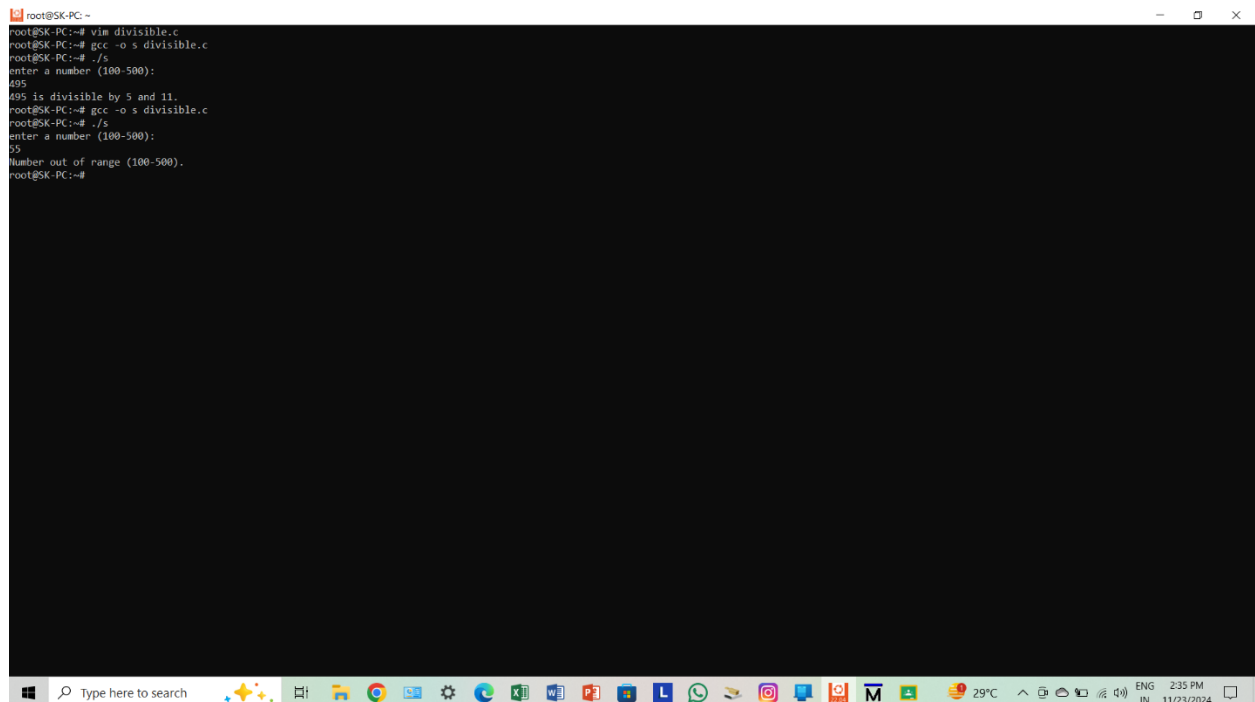
    printf("Number out of range (100-500).\n");

}

return 0;

}
```

OUTPUT



```
root@SK-PC: ~  
root@SK-PC:~# vim divisible.c  
root@SK-PC:~# gcc -o s divisible.c  
root@SK-PC:~# ./s  
enter a number (100-500):  
495  
495 is divisible by 5 and 11.  
root@SK-PC:~# gcc -o s divisible.c  
root@SK-PC:~# ./s  
enter a number (100-500):  
55  
Number out of range (100-500).  
root@SK-PC:~#
```


16. Write a C program to check whether a number is even or odd.

```
#include <stdio.h>

int main()
{
    int n;

    printf("enter a number: ");

    scanf("%d", &n);

    if(n%2==0)

        printf("%d is an even numbr.\n", n);

    else

        printf("%d is an odd number.\n", n);

    return 0;
}
```

OUTPUT

```
root@SK-PC: ~  
root@SK-PC:~# vim eo.c  
root@SK-PC:~# gcc -o s eo.c  
root@SK-PC:~# ./s  
enter a number:  
8  
8 is an even number.  
root@SK-PC:~# gcc -o s eo.c  
root@SK-PC:~# ./s  
enter a number: 5  
5 is an odd number.  
root@SK-PC:~#
```

17. Write a C program to check whether a year is a leap year or not.

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int y;
```

```
    printf("enter the year: ");
```

```
    scanf("%d", &y);
```

```
    if((y%4==0 && y%100!=0) || (y%400==0))
```

```
        printf("%d is a leap year.\n", y);
```

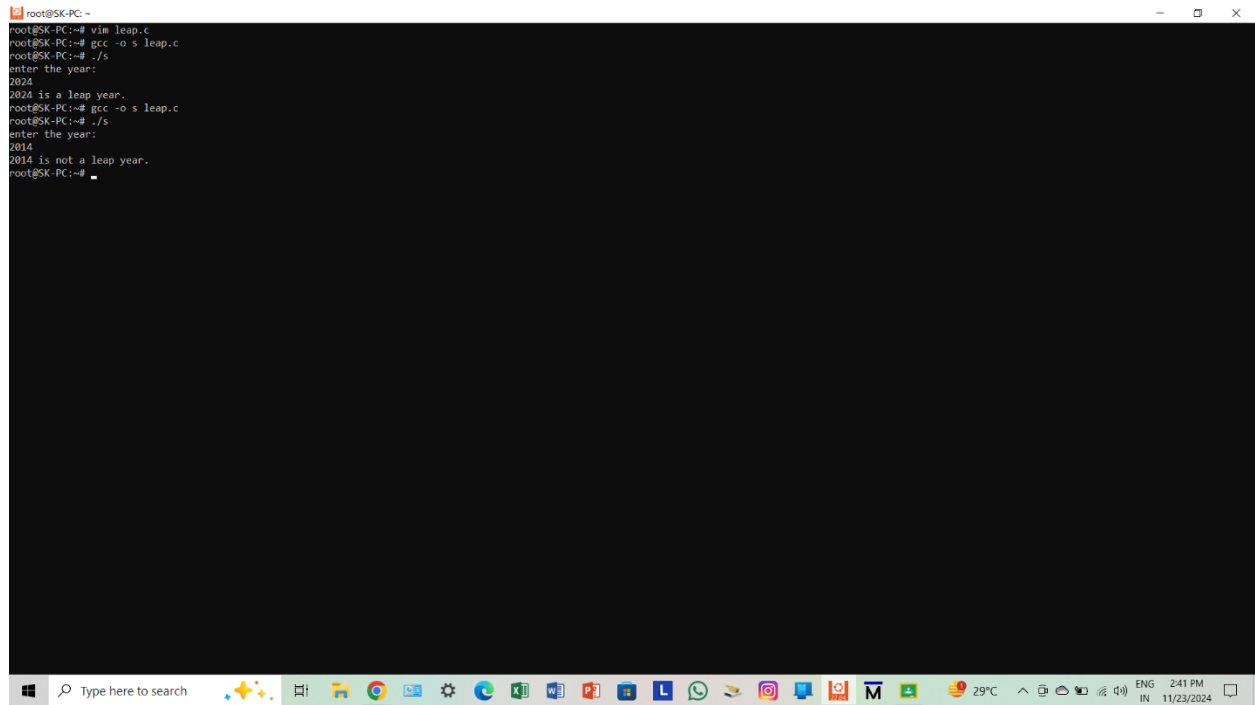
```
    else
```

```
    printf("%d is not a leap year.\n", y);

return 0;

}
```

OUTPUT



The screenshot shows a terminal window with the following text:

```
root@SK-PC: ~  
root@SK-PC:~# vim leap.c  
root@SK-PC:~# gcc -o s leap.c  
root@SK-PC:~# ./s  
enter the year:  
2024  
2024 is a leap year.  
root@SK-PC:~# gcc -o s leap.c  
root@SK-PC:~# ./s  
enter the year:  
2014  
2014 is not a leap year.  
root@SK-PC:~#
```

The terminal window is titled "root@SK-PC: ~" and has standard window controls (minimize, maximize, close) in the top right corner. The Windows taskbar is visible at the bottom, showing the search bar, task view, and various application icons. The system tray on the right indicates a temperature of 29°C, the date 11/23/2024, and the time 2:41 PM.

18. Write a C program to check whether a character is alphabet or not.

```
#include <stdio.h>
```

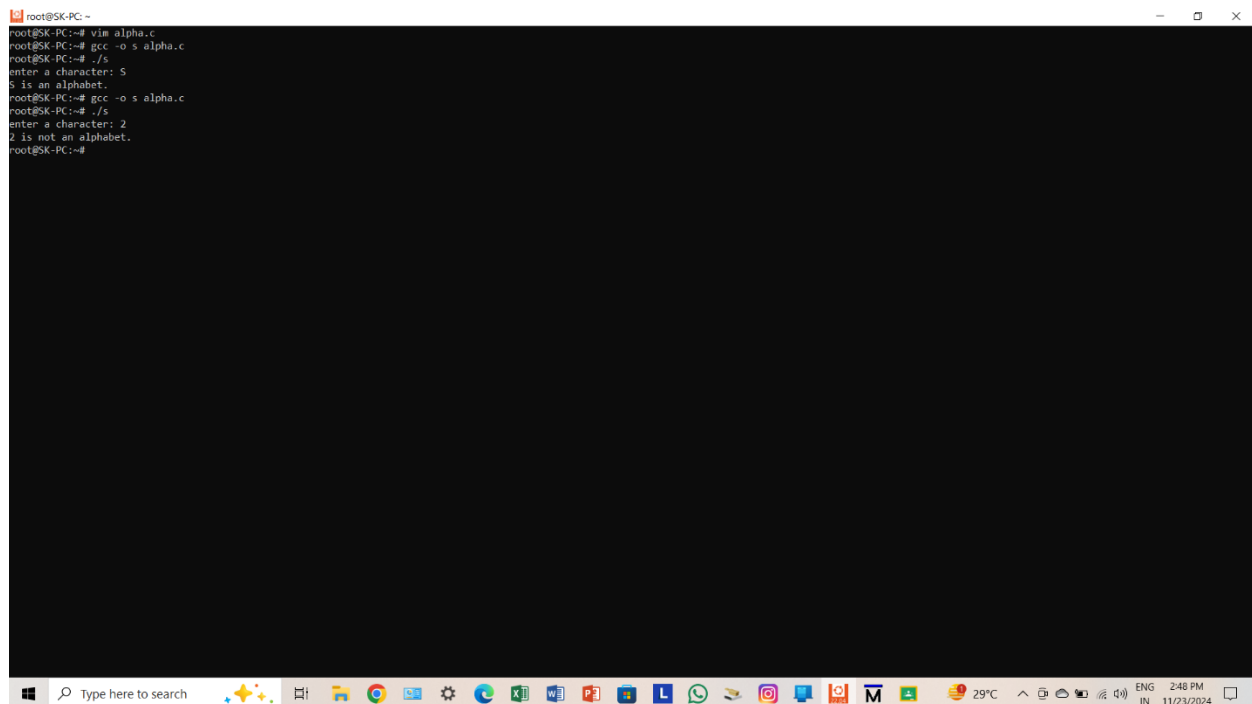
```
int main()
```

```
{
```

```
    char chr;
```

```
printf("enter a character: ");  
  
scanf("%c", &chr);  
  
if((chr>= 'A' && chr<= 'Z') || (chr>= 'a' && chr<= 'z'))  
  
    printf("%c is an alphabet.\n", chr);  
  
else  
  
    printf("%c is not an alphabet.\n", chr);  
  
return 0;  
  
}
```

OUTPUT



```
root@SK-PC: ~  
root@SK-PC:~# vim alpha.c  
root@SK-PC:~# gcc -o s alpha.c  
root@SK-PC:~# ./s  
enter a character: S  
S is an alphabet.  
root@SK-PC:~# gcc -o s alpha.c  
root@SK-PC:~# ./s  
enter a character: Z  
Z is not an alphabet.  
root@SK-PC:~#
```

19. Write a C program to input any alphabet and check whether it is a vowel or consonant.

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    char ch;
```

```
    printf("enter an alphabet: ");
```

```
    scanf("%c", &ch);
```

```
    if((ch >= 'A' && ch <= 'Z') || (ch >= 'a' && ch <= 'z'))
```

```
    {
```

```
        if(ch== 'A' || ch== 'E' || ch== 'I' || ch== 'O' || ch== 'U' || ch== 'a' || ch==  
'e' || ch== 'i' || ch== 'o' || ch== 'u')
```

```
            printf("%c is a vowel.\n", ch);
```

```
        else
```

```
            printf("%c is a consonant.\n", ch);
```

```
    }
```

```
else
```

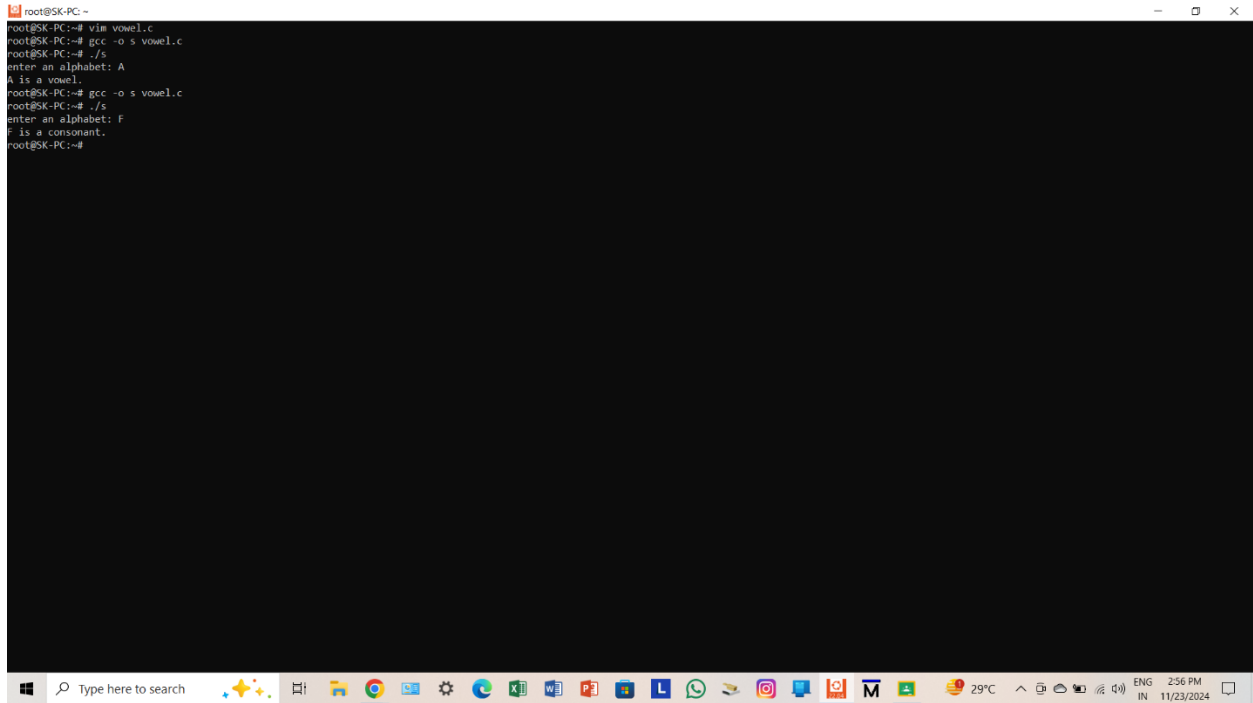
```
{
```

```
    printf("%c is not an alphabet.\n", ch);
```

```
}
```

```
    return 0;
}
```

OUTPUT



```
root@SK-PC: ~  
root@SK-PC:~# vim vowel.c  
root@SK-PC:~# gcc -o s vowel.c  
root@SK-PC:~# ./s  
enter an alphabet: A  
A is a vowel.  
root@SK-PC:~# gcc -o s vowel.c  
root@SK-PC:~# ./s  
enter an alphabet: F  
F is a consonant.  
root@SK-PC:~#
```

20. Write a C program to input any character and check whether it is an alphabet, digit, or special character.

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    char c;
```

```
    printf("enter a character: ");
```

```
scanf("%c", &c);

if((c >= 'A' && c <= 'Z') || (c >= 'a' && c <= 'z'))

    printf("%c is an alphabet.\n", c);

else if(c >= '0' && c <= '9')

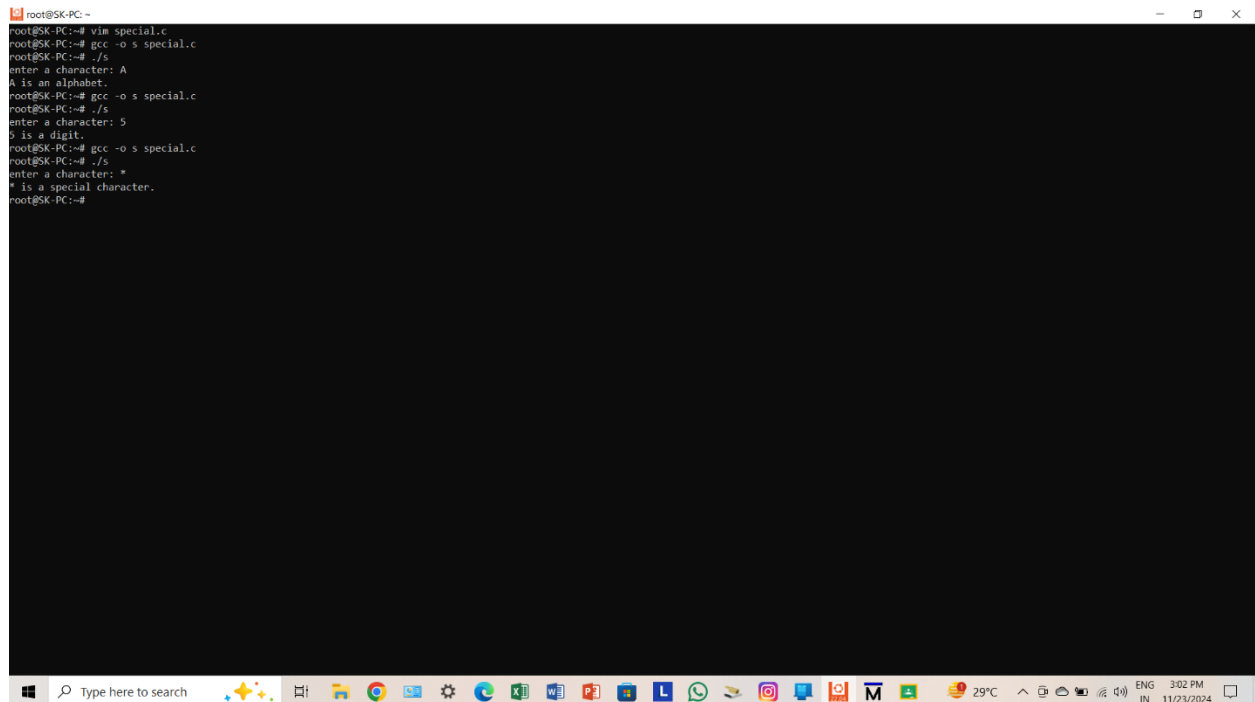
    printf("%c is a digit.\n", c);

else

    printf("%c is a special character.\n", c);

return 0;
}
```

OUTPUT



```
root@SK-PC: ~  
root@SK-PC:~# vim special.c  
root@SK-PC:~# gcc -o s special.c  
root@SK-PC:~# ./s  
enter a character: A  
A is an alphabet.  
root@SK-PC:~# gcc -o s special.c  
root@SK-PC:~# ./s  
enter a character: 5  
5 is a digit.  
root@SK-PC:~# gcc -o s special.c  
root@SK-PC:~# ./s  
enter a character: *  
* is a special character.  
root@SK-PC:~#
```

The screenshot shows a Linux terminal window with a black background. The user is at the root prompt on a machine named SK-PC. They have edited a file named special.c using vim, compiled it with gcc to create an executable named s, and then run it three times. Each time, they enter a character, and the program outputs a message indicating whether the character is an alphabet, a digit, or a special character. The first input is 'A', which is an alphabet. The second input is '5', which is a digit. The third input is '*', which is a special character. The terminal window has standard Linux window controls at the top right. At the bottom, there is a taskbar with various application icons, a search bar, and system status information including temperature (29°C), time (3:02 PM), and date (11/23/2024).