غلام اسحٰق خان انسٹیٹیوٹ آف انجینئرنگ سائنس و ٹیکنالوجی
Ghulam ishaq khan institute of engineering sciences and technology

NAME: MUHAMMAD HARIS

CLASS: CS-SECTION H.

REG NO: 2023428

Q No 01(CLO # 3, PLO #1)

An integer is said to be prime if it's divisible by only 1 and itself. For example, 2, 3, 5, and 7 are prime, but 4, 6, 8, and 9 are not.

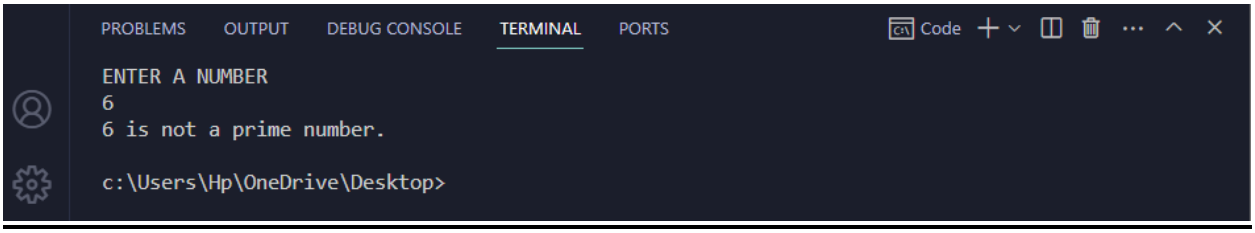
i. Write a function that **determines** whether a number is a prime.
ii. Use this function in a program that **identifies** and prints all the prime numbers between 2 and 10,000. How many of these numbers do you really have to test before being sure that you've found all the primes?

CODE (PART 1):

```
#include <iostream>
using namespace std;

bool prime(int num) //intiallizong the function that will determine if the no is
prime or not.
{
    if(num<=1)
    {
        return false;
    }
// the for loop ahead would start from two and would only check the first 2
numbers between I and the user input number as it would be sufficient to see if
the remainder is zero to make the user input non-prime number.
    for (int i=2; i<=num/2; i++)
    {
    if (num%i==0)
    {
```

```
    return false;
    }
    }
return true;
}
int main()
{
    int num;
    cout<<"ENTER A NUMBER"<<endl;
    cin>>num;

if(prime(num))
{
    cout<<num<<" is a prime number."<<endl;
}
else
{
cout<<num<<" is not a prime number."<<endl;
}
return 0;
}
```

SOLUTION (PART 1):



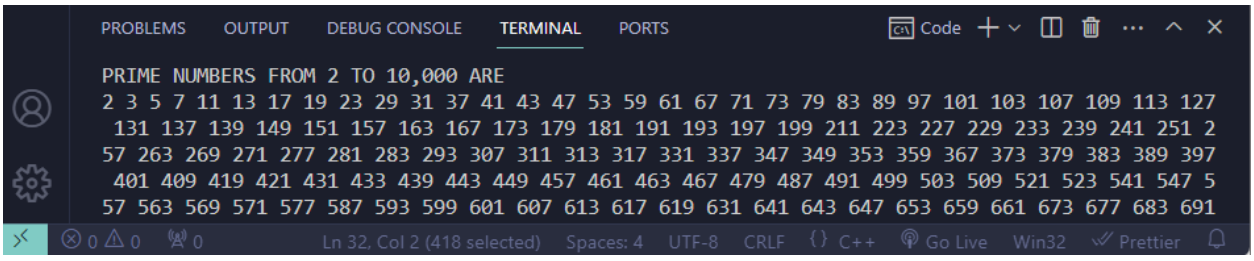
CODE (PART 2):

In order to find the all the possible prime numbers, we have to check starting from 2 to half of the input no. to be sure of printing all the required prime numbers.

```
    for (int i=2; i<=num/2; i++)
    {
    if (num%i==0)
    {
```

```cpp
#include <iostream>
using namespace std;

bool prime(int num)
{
    if(num<=1)
    {
        return false;
    }
    for (int i=2; i<=num/2; i++)
    {
    if (num%i==0)
    {
    return false;
    }
    }
return true;
}
int main()
{
    cout<<"PRIME NUMBERS FROM 2 TO 10,000 ARE"<<endl;

for(int i=2; i<=10000; i++)
{
    if(prime(i))
    {
    cout<<i<<" ";
}
}
cout<<endl;
return 0;
}
```

SOLUTION (PART 2):



Q No 02 (CLO # 3, PLO #1)

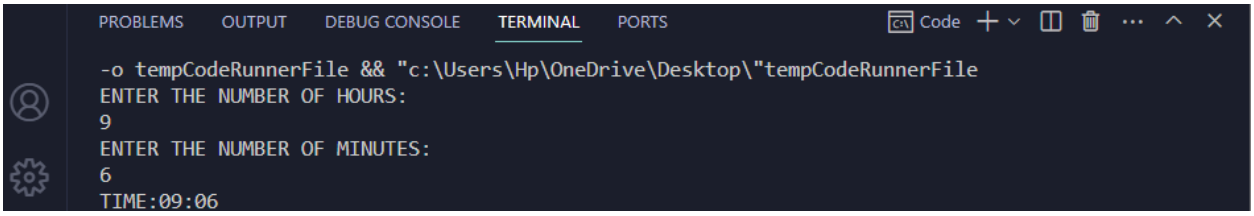
Implement a clock by writing a C++ program that asks the user to enter an hour value and a minute value. The main () function should then pass these two values to a type void function that **displays** the two values in the format shown in the following sample run:
Enter the number of hours: 9
Enter the number of minutes: 28
Time: 9:28

CODE:

```
#include <iostream>
using namespace std;

void displayTime(int hours, int minutes)
//initialized a function for displaying the time
{
    cout << "TIME:";
    if(hours<10)
// if the time is less than 10 then putting a zero before its initial.
    {
    cout<<"0";
    }
    cout<< hours << ":";
// similarly putting a zero if the minutes are a single digit number.
    if (minutes < 10)
    {
        cout << "0";
    }
    cout << minutes << endl;
}

int main()
{
    int hours, minutes;
```

```
    cout << "ENTER THE NUMBER OF HOURS: " << endl;
    cin >> hours;

    cout << "ENTER THE NUMBER OF MINUTES: " << endl;
    cin >> minutes;

    displayTime(hours, minutes);

    return 0;
}
```

SOLUTION:



Q No 03 (CLO # 3, PLO #1)

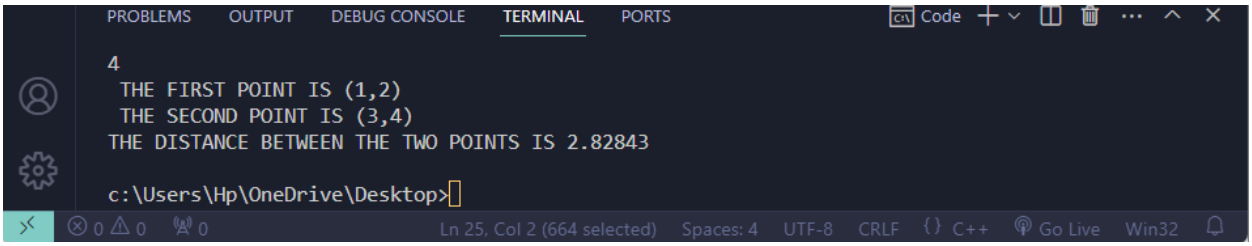
Write a function that **computes** the distance between two points *(x1, y1)* and *(x2, y2)*. All numbers and return values should be of type double.

CODE:

```
#include <iostream>
#include <cmath>
using namespace std;


double distance(double x1, double x2, double y1, double y2) //initializing the
funct to make the formula code of the finding the distance between 2 points.
{
    double answer= sqrt(pow(x2-x1, 2)+ pow(y2-y1, 2));
    return answer;
}

int main()
{
    double x1, x2, y2, y1;
    cout<<"ENTER THE COORDINATES OF FIRST POINT I.E (X1, Y1)"<<endl;
    cin>>x1>>y1;
```

```
     cout<<"ENTER THE COORDINATES OF SECOND POINT I.E (X2, Y2)"<<endl;
    cin>>x2>>y2;
cout<<" THE FIRST POINT IS "<<"("<<x1<<","<<y1<<")"<<endl;
cout<<" THE SECOND POINT IS "<<"("<<x2<<","<<y2<<")"<<endl;
    cout<<"THE DISTANCE BETWEEN THE TWO POINTS IS "<<distance(x1,x2,y1,y2);
    cout<<endl;

    return 0;
}
```

SOLUTION:



Q.No.04 (CLO # 3, PLO #1)

Write a program that plays the game of "guess the number" as follows: Your program chooses the number to be guessed by selecting an integer at random in the range 1 to 1000. The program then types:

```
I have a number between 1 and 1000.
Can you guess my number?
Please type your first guess.
```

The player then types a first guess. The program responds with one of the following:

1. Excellent! You guessed the number!
Would you like to play again (y or n)?
2. Too low. Try again.
3. Too high. Try again.

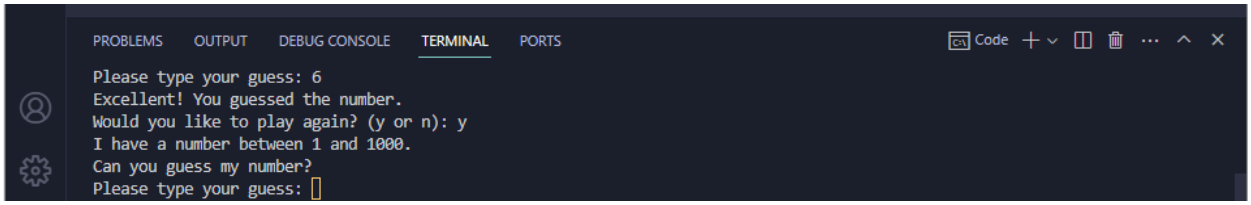
If the player's guess is incorrect, your program should loop until the player finally gets the number right. Your program should keep telling the player Too high or Too low to help the player "zero in" on the correct answer.

CODE:

```
#include <iostream>
#include <cstdlib>
#include <ctime>

using namespace std;

int main() {
    srand(time(0)); // for setting a unique random number everytime the program
runs.
      char playagain;

    do { //using do while loop, the do will first always enter in another do and
ask for the guesses until the correct guess is reached after which it will ask
the user if it want's to repeat the action of the code I.e the game.
        int a = 1 + rand() % 100;
        int guess;

        cout<<"I have a number between 1 and 100."<<endl;
        cout<<"Can you guess my number?"<<endl;

        do
{
            cout<<"Please type your guess:";
            cin>>guess;

            if(guess == a) {
                cout<<"Excellent! You guessed the number."<<endl;
            } else if (guess < a)
{
                cout<<"Too low. Try again."<<endl;
            } else {
                cout<<"Too high. Try again."<<endl;
            }

        }while(guess != a);

        cout<<"Would you like to play again? (y or n): ";
        cin>>playagain;

    } while(playagain=='y' || playagain=='Y');

    cout<<"Thanks for playing! Goodbye."<<endl;
```

```
    return 0;
}
```

SOLUTION:



Q.No.05 (CLO # 3, PLO #1)

The Fibonacci series

0, 1, 1, 2, 3, 5, 8, 13, 21, …

begins with the terms 0 and 1 and has the property that each succeeding term is the sum of the two preceding terms.

a) Write a non-recursive function Fibonacci (n) that calculates the nth Fibonacci number.
b) Modify the program of part a) to use double instead of int to calculate and return Fibonacci numbers and use this modified program to repeat part b).

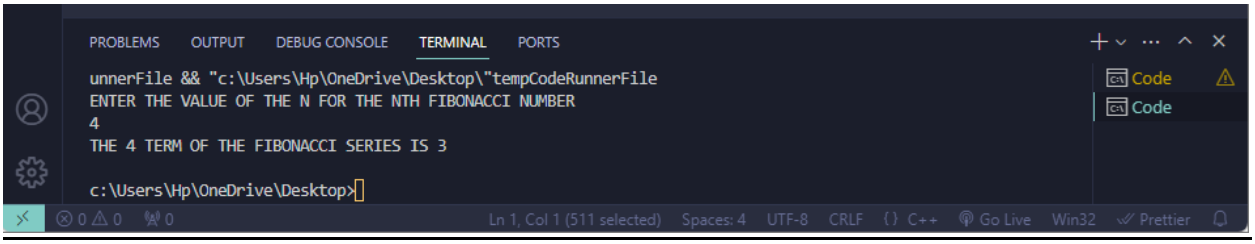
NOTE: 0 IS AT 0^{TH} TERM AND 1 IS AT 1^{ST} TERM POSITION(AND SO ON).

CODE (PART 1):

```
#include <iostream>
using namespace std;

int fib(int n)
{
    int a=0;
    int b=1;
int nexterm=0;
```

```
for(int i=2; i<=n; i++) //using loop to find the nextterm and starting it from 2 as the the first 2 terms are already known, also renewing the terms so that if the user asks for an nth term, the program can sum the last 2 known numbers to give the nth term.
{
    nexterm=a+b;
    a=b;
    b=nexterm;
}
if(n==0)
{return 0;}
else
{return b;}
}
int main()
{
    int n;
    cout<<"ENTER THE VALUE OF THE N FOR THE NTH FIBONACCI NUMBER"<<endl;
    cin>>n;
    if(n<0)
    {
        cout<<"INVALID INPUT. ENTER A NON-NEGATIVE INTEGER"<<endl;
    }
    cout<<"THE "<<n<<" TERM OF THE FIBONACCI SERIES IS "<<fib(n)<<endl;
}
```

SOLUTION(PART 1):

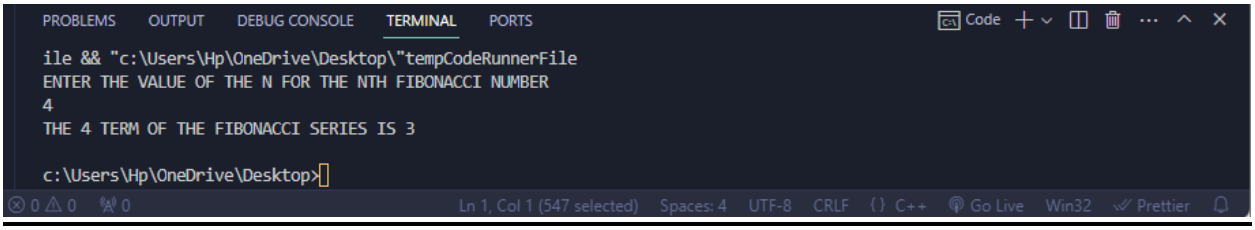


CODE (PART 2):

```
#include <iostream>
using namespace std;

double fib(double n)
{
    if (n==0)
    {
```

```
    double a=0.0;
    return a;
    }
    if (n==1)
    {
// for the given Fibonacci series the first 2 values are give and if the user
asks for them they will be simply outputted.
    double b=1.0;
    return b;
    }
// how ever if the user asks for any other term, the recursive funct will run to
first achieve the 2 consecutive number just before that specific term and
eventually sum them to give the output.
 return fib(n-2)+ fib(n-1);
}
int main()
{
    double n;
    cout<<"ENTER THE VALUE OF THE N FOR THE NTH FIBONACCI NUMBER"<<endl;
    cin>>n;
    if(n<0)
    {
        cout<<"INVALID INPUT. ENTER A NON-NEGATIVE double"<<endl;
    }
    else
    {
    cout<<"THE "<<n<<" TERM OF THE FIBONACCI SERIES IS "<<fib(n)<<endl;
return 0.0;
}
}
```

SOLUTION(PART2):



Q.No.06 (CLO # 3, PLO #1)

a) Create a C++ recursive function to calculate the sum of digits for a given positive integer. The sum of digits is obtained by adding up each digit in the number. For example, the sum of digits for 123 should be 6 (1 + 2 + 3). Your function should handle the base case and demonstrate the recursive nature of the solution. Then write the main function in which the user should input 3-digit number and call the recursive function to demonstrate the results of any three-digit number.

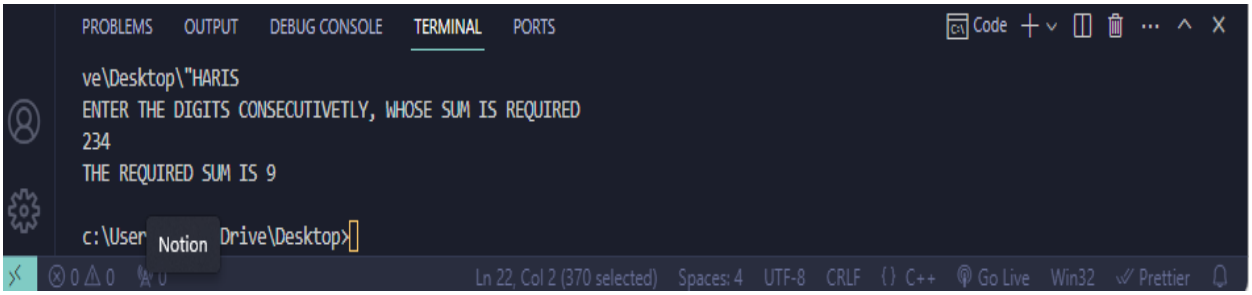
b) Design a recursive function in C++ to check if a given number, is a **palindrome.**

(same number even the digits are reversed). Explain the base case and recursive case for this function.)

CODE (PART 1):

```
#include <iostream>
using namespace std;

int sumofdigits(int n)
{
//doing the task of finding the sum of the number for which the digits are chosen
from the number given, we output the user input number if it is less than 10 as
there would be nothing to sum up with.
    if(n<10)
    {
        return n;
    }
// that was above the base case and thus if the 2 digit or more digit no. is give
the program runs by taking the last digit of that number first and recursively
calling itself reducing the number by one digit, i.e again taking its last digit
and adding with the previous. This practice will eventually run until the base
case is reached.
    else
    {
    return n%10+sumofdigits(n/10);
    }
}
int main()
{
    int n;
    cout<<"ENTER THE DIGITS CONSECUTIVETLY, WHOSE SUM IS REQUIRED"<<endl;
    cin>>n;
    cout<<"THE REQUIRED SUM IS "<<sumofdigits(n)<<endl;
    return 0;
}
```

SOLUTION(PART 1):



CODE (PART 2):

```cpp
#include <iostream>
using namespace std;

int palindrome(int num, int &originalnum, int &reversenum)
{
    if (num == 0)
    {
        return originalnum == reversenum;
    }
    else
    {
        int lastdigit = num % 10;

        reversenum = reversenum * 10 + lastdigit;
        num = num / 10;
        return palindrome(num, originalnum, reversenum);
    }
}

int main()
{
    int number;
    cout << "ENTER A NUMBER TO CHECK IF IT IS PALINDROME" << endl;
    cin >> number;

    int originalnum = number;

    int reversenumber = 0;

    if (palindrome(number, originalnum, reversenumber))
    {
```

```
        cout << originalnum << " is a palindrome." << endl;
    }
    else
    {
        cout << originalnum << " is not a palindrome." << endl;
    }
    return 0;
}
```

SOLUTION(PART 2):

We have set the base case equal to zero as it checkes that all the digits have been processed in the original number, when it is true the funct will output the original number which is now the reversed number.

The recursive function is defined by the else block. In this case the function is not in the base case meaning the number is not zero. It will extract the last digit of the number and update the reverse number, it also updates the number in use by dividing it by 10 and removing the lst digit as it is already moved as first digit in the reversed number.
The function would recursively call itself until the base case is satisfied.