

Experiment Name :

Friend Function and Friend Class in C++.

Objectives:

- Familiarize with friend class and function in C++ .
- Solve some problems using friend function.

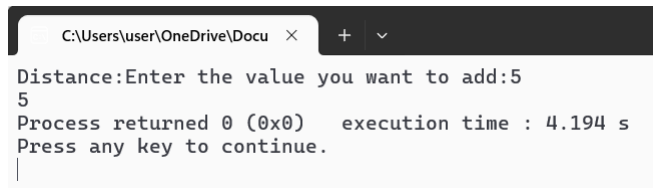
Example-1 :

A C++ program to demonstrate the use of friend function.

Input:

```
#include<iostream>
using namespace std;
class Distance{
private:
    int meter;
    int value;
    friend int addvalue(Distance);
public:
    Distance()
    {
        meter=0;
    }
};
int addvalue(Distance d){
    cout<<"Enter the value you want to add:";
    cin>>d.value; //d=receiving argument
    return d.meter+d.value;
}
int main()
{
    Distance D;
    cout<<"Distance:"<<addvalue(D);
    return 0;
}
```

Output :



```
C:\Users\user\OneDrive\Docu × + v
Distance:Enter the value you want to add:5
5
Process returned 0 (0x0) execution time : 4.194 s
Press any key to continue.
|
```

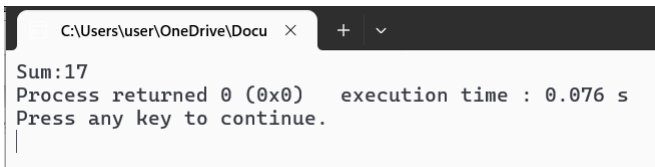
Example-2 :

Add members of two different classes using friend functions.

Input:

```
#include<iostream>
using namespace std;
class Class2;
class Class1 {
private:
    int num1;
public:
    Class1()
    {
        num1=16;
    }
    friend int add(Class1, Class2);
};
class Class2{
private:
    int num2;
public:
    Class2()
    {
        num2=1;
    }
    friend int add(Class1, Class2);
};
int add(Class1 object1, Class2 object2){
    return (object1.num1+object2.num2);
}
int main()
{
    Class1 object1;
    Class2 object2;
    cout<<"Sum:"<<add(object1, object2);
    return 0;
}
```

Output :



```
C:\Users\user\OneDrive\Docu × + ▾
Sum:17
Process returned 0 (0x0)   execution time : 0.076 s
Press any key to continue.
|
```

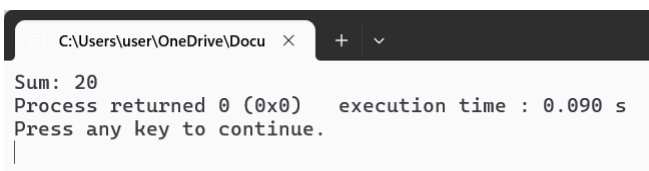
Example-3 :

A C++ program to demonstrate the working of friend class.

Input:

```
#include <iostream>
using namespace std;
class ClassB;
class ClassA {
private:
    int numA;
    friend class ClassB;
public:
    ClassA(){
        numA = 16;
    }
};
class ClassB {
private:
    int numB;
public:
    ClassB(){
        numB = 4;
    }
    int add() {
        ClassA objectA;
        return objectA.numA + numB;
    }
};
int main() {
    ClassB objectB;
    cout << "Sum: " << objectB.add();
    return 0;
}
```

Output :



```
C:\Users\user\OneDrive\Docu × + ▾
Sum: 20
Process returned 0 (0x0)   execution time : 0.090 s
Press any key to continue.
|
```

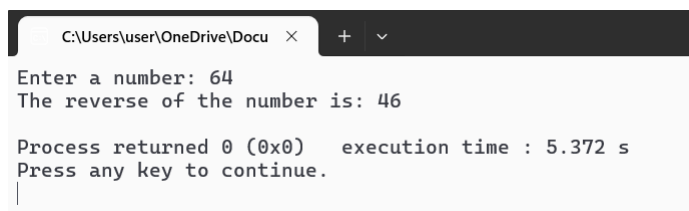
Practice Exercise: 1

Write a C++ Program to display the reverse of a number using the Friend function.

Input:

```
#include <iostream>
using namespace std;
class Number {
private:
    int value;
public:
    // Constructor to initialize the number
    Number(int v) : value(v) {}
    // Friend function declaration
    friend int reverseNumber(const Number& num);
};
int reverseNumber(const Number& num) {
    int reversed = 0;
    int temp = num.value;
    while (temp != 0) {
        int digit = temp % 10;
        reversed = reversed * 10 + digit;
        temp /= 10;
    }
    return reversed;
}
int main() {
    int num;
    cout << "Enter a number: ";
    cin >> num;
    Number numberObj(num);
    cout << "The reverse of the number is: " << reverseNumber(numberObj) << endl;
    return 0;
}
```

Output :

A screenshot of a C++ program execution window. The window title bar shows the file path 'C:\Users\user\OneDrive\Docu' and standard window controls. The main content area displays the program's output: 'Enter a number: 64' followed by 'The reverse of the number is: 46'. At the bottom, a status bar indicates 'Process returned 0 (0x0) execution time : 5.372 s' and prompts the user to 'Press any key to continue.'

```
C:\Users\user\OneDrive\Docu  ×  +  ▾
Enter a number: 64
The reverse of the number is: 46

Process returned 0 (0x0)   execution time : 5.372 s
Press any key to continue.
|
```

Practice Exercise: 2

Write a C++ program to find the number and sum of all integer between 100 and 200 which are divisible by 11 with friend function.

Input:

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

```

class Range {
private:
    int start, end;
public:
    // Constructor to initialize range
    Range(int s, int e) : start(s), end(e) {}
    // Friend function declaration
    friend void findDivisibleBy11(const Range& r);
};

void findDivisibleBy11(const Range& r) {
    int sum = 0;
    cout << "Numbers between " << r.start << " and " << r.end << " divisibl
    for (int i = r.start; i <= r.end; ++i) {
        if (i % 11 == 0) {
            cout << i << " ";
            sum += i;
        }
    }
    cout << "\nSum of these numbers: " << sum << endl;
}

int main() {
    Range range(100, 200);
    findDivisibleBy11(range);
    return 0;
}

```

Output :

```

C:\Users\user\OneDrive\Docu  ×  +  ▾
Numbers between 100 and 200 divisible by 11 are: 110 121 132 143 154 165 176 187 198
Sum of these numbers: 1386

Process returned 0 (0x0)   execution time : 0.065 s
Press any key to continue.

```

Practice Exercise: 3

Write a program in C++ to Check Whether a Number can be expressed as Sum of Two Prime Numbers using the friend function.

Input:

```

#include <iostream>
using namespace std;
class Number {
private:
    int value;
public:
    // Constructor to initialize the number
    Number(int v) : value(v) {}
    // Friend function declaration
    friend void checkSumOfTwoPrimes(Number num);
    // Utility function to check if a number is prime
    static bool isPrime(int n) {

```

```

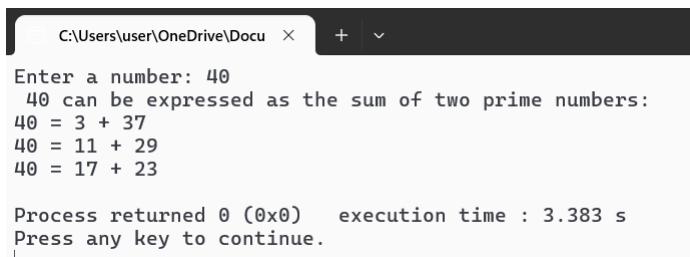
        if (n <= 1) return false;
        for (int i = 2; i <= sqrt(n); ++i) {
            if (n % i == 0) return false;
        }
        return true;
    }
};

void checkSumOfTwoPrimes(Number num) {
    int n = num.value;
    bool found = false;
    cout << " " << n << "can be expressed as the sum of two prime numbers:\n";
    for (int i = 2; i <= n / 2; ++i) {
        int complement = n - i;
        if (Number::isPrime(i) && Number::isPrime(complement)) {
            cout << n << " = " << i << " + " << complement << endl;
            found = true;
        }
    }
    if (!found) {
        cout << n << "cannot be expressed as the sum of two prime numbers.\n";
    }
}

int main() {
    int num;
    cout << "Enter a number: ";
    cin >> num;
    Number numberObj(num);
    checkSumOfTwoPrimes(numberObj);
    return 0;
}

```

Output :



```

C:\Users\user\OneDrive\Docu  ×  +  ▾
Enter a number: 40
40 can be expressed as the sum of two prime numbers:
40 = 3 + 37
40 = 11 + 29
40 = 17 + 23

Process returned 0 (0x0)   execution time : 3.383 s
Press any key to continue.

```

Discussion:

In this experiment above, friend function and friend class was introduced. Here, the friend function effectively demonstrated controlled access to a class's private member to implement specific functionality such as checking the sum of two primes, displaying reverse digits etc. It is observed that when specific functions need access to private data, friend function was used and friend class granted access to all members of a class. While solving the problems, I faced a little difficulty when declaring a friend class. But after implementing part by part, the concept was understood and so fixing the errors, the problems were solved as well.