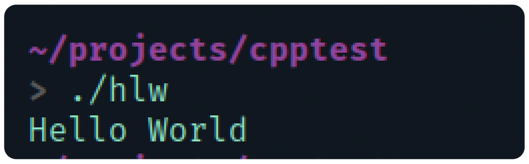
Simple Programs :

Practical - 1 : Write a program to display "Hello World !"

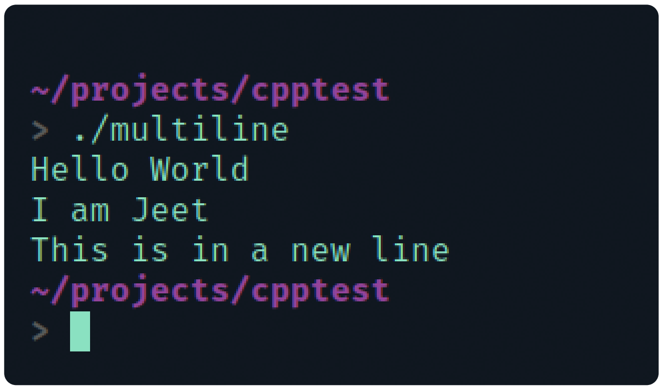
```cpp
#include <iostream>
using namespace std;

int main (){
cout<<"Hello World";

}
```



```
~/projects/cpptest
> ./hlw
Hello World
```

Practical - 2 : Write a program to insert new lines to display multiple statements: ["Hello World !","I am Jeet"]

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

int main (){
cout<<"Hello World"<<endl<<"I am Jeet";
cout<<"\nThis is in a new line";

}
```



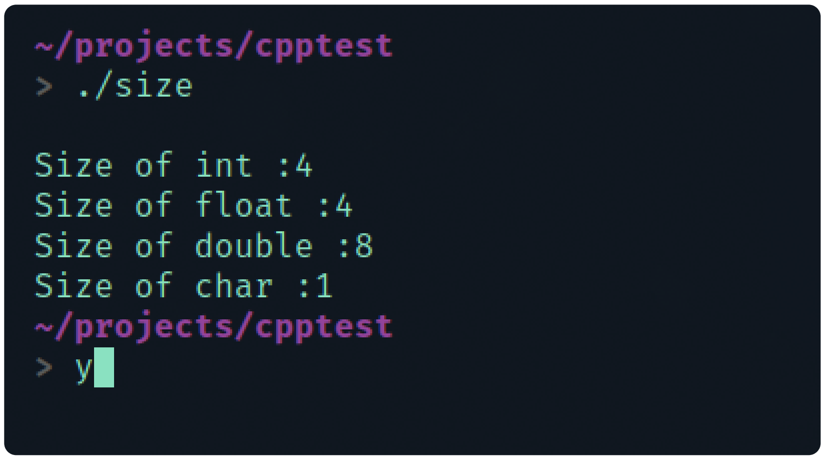
```
~/projects/cpptest
> ./multiline
Hello World
I am Jeet
This is in a new line
~/projects/cpptest
> 
```

Practical - 3 : Write a program to write single line and multi-line comments.

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

int main (){

// This is a single line comment

/*
This is a multi line comment
This is written using the "/*  ... "
*/
}
```

Practical - 4 : Write a program to display size of [int, float, double, char] in your system.

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

int main (){
cout<<"Size of int :"<<sizeof(int);
cout<<"Size of float :"<<sizeof(float);
cout<<"Size of double :"<<sizeof(double);
cout<<"Size of char :"<<sizeof(char);
}
```

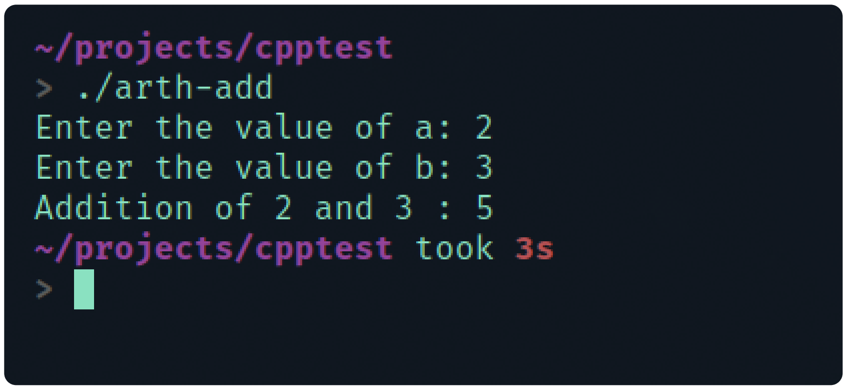


```
~/projects/cpptest
> ./size

Size of int :4
Size of float :4
Size of double :8
Size of char :1
~/projects/cpptest
> y
```

Class Programs :

Practical - 1 : Write a program to perform addition using arithematic addition.

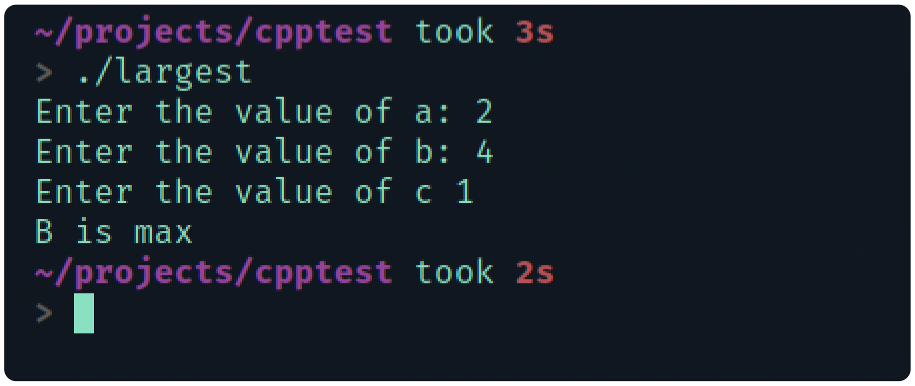
```cpp
#include <iostream>
using namespace std;

class Arithematic {
private:
        int a,b;

public:
        void add(int a, int b){
                cout<<"Addition of "<<a<<" and "<<b<<" : "<<a+b;
        }
};

int main (){
        int a,b;

        cout<<"Enter the value of a: ";
        cin>>a;
        cout<<"Enter the value of b: ";
        cin>>b;
        Arithematic a1;
        a1.add(a,b);
        return 0;
}
```



```
~/projects/cpptest
> ./arth-add
Enter the value of a: 2
Enter the value of b: 3
Addition of 2 and 3 : 5
~/projects/cpptest took 3s
> 
```

Practical - 2 : Write a program to find the largest number among Three numbers.

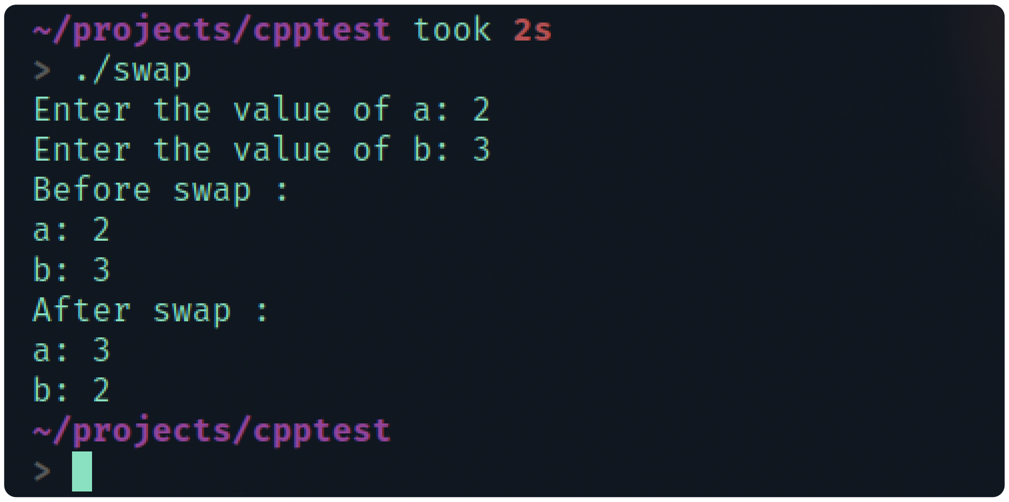
```cpp
#include <iostream>
using namespace std;

class Arithematic {
private:
        int a,b,c;

public:
        void max(int a, int b, int c){
                a>b? a>c? cout<<"A is max" : cout<<"C is max" : b>c?
cout<<"B is max" : cout<<"C is max";
        }
};

int main (){
        int a,b,c;
        cout<<"Enter the value of a: ";
        cin>>a;
        cout<<"Enter the value of b: ";
        cin>>b;
        cout<<"Enter the value of c ";
        cin>>c;
        Arithematic a1;
        a1.max(a,b,c);
        return 0;
}
```



```
~/projects/cpptest took 3s
> ./largest
Enter the value of a: 2
Enter the value of b: 4
Enter the value of c 1
B is max
~/projects/cpptest took 2s
>
```

Practical - 3 : Write a program to swap two numbers.

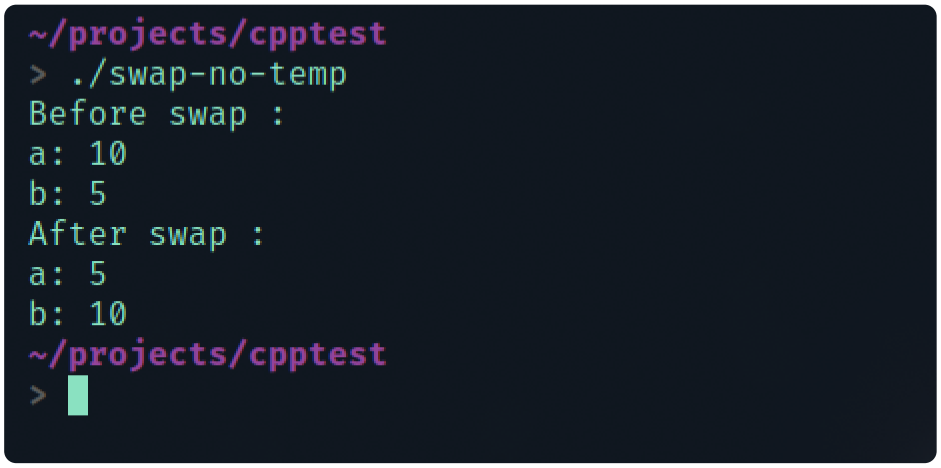
```
#include <iostream>
using namespace std;

class Operations {
public:
void swap(int a, int b){
        int temp;
        temp = a;
        a = b;
        b = temp;

        cout<<"After swap : \na: "<<a<<"\nb: "<<b;
        }
};

int main (){
        int a,b;

        cout<<"Enter the value of a: ";
        cin>>a;
        cout<<"Enter the value of b: ";
        cin>>b;

        cout<<"Before swap : \na: "<<a<<"\nb: "<<b<<endl;

        Operations o1;
        o1.swap(a,b);
        return 0;
}
```



```
~/projects/cpptest took 2s
> ./swap
Enter the value of a: 2
Enter the value of b: 3
Before swap :
a: 2
b: 3
After swap :
a: 3
b: 2
~/projects/cpptest
>
```

Practical - 4 : Write a program to swap two numbers without using temporary numbers.

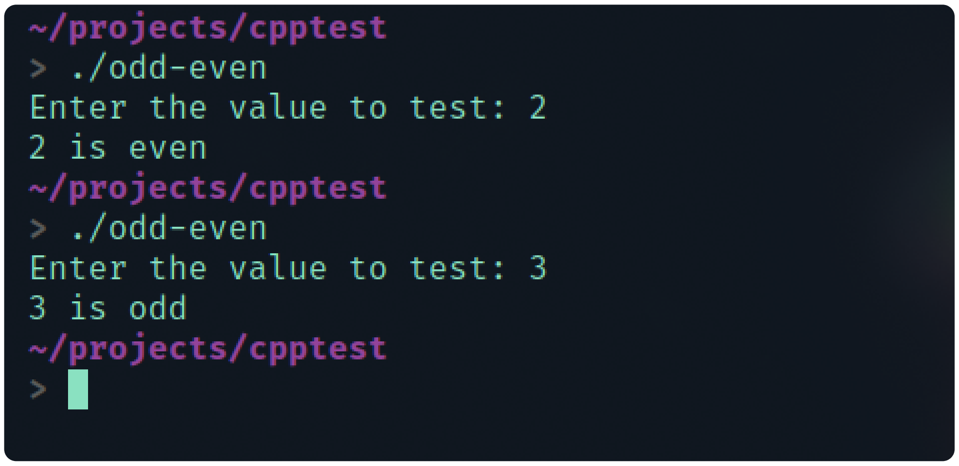
```
#include <iostream>
using namespace std;

class Operations {
private:
        int a = 10,b = 5;

public:
        void swap(int a, int b){

        cout<<"Before swap : \na: "<<a<<"\nb: "<<b<<endl;
    a = a+b;
    b = a-b;
    a = a-b;
        cout<<"After swap : \na: "<<a<<"\nb: "<<b;
        }
};

int main (){
    Operations o1;
        o1.swap(10,5);

        return 0;
}
```



```
~/projects/cpptest
> ./swap-no-temp
Before swap :
a: 10
b: 5
After swap :
a: 5
b: 10
~/projects/cpptest
>
```

Practical - 5 : Write a program to find if number is odd or even.

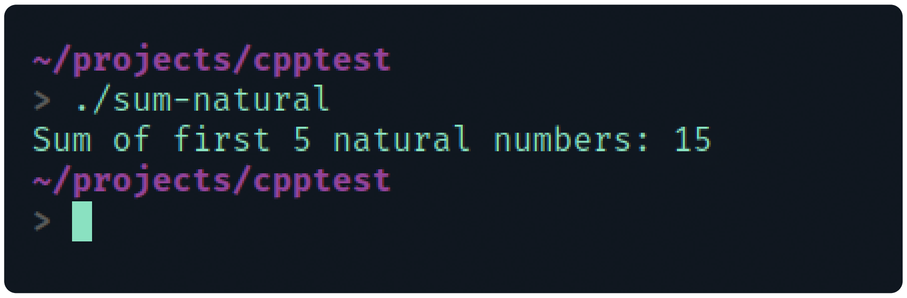
```
#include <iostream>
using namespace std;

class Operations {
private:
        int a;

public:
        void isEven(int a){
        (a%2==0)?cout<<a<<" is even":cout<<a<<" is odd";
        }
};

int main (){
        int a;
        Operations o1;
        cout<<"Enter the value to test: ";
        cin>>a;
        o1.isEven(a);

        return 0;
}
```



```
~/projects/cpptest
> ./odd-even
Enter the value to test: 2
2 is even
~/projects/cpptest
> ./odd-even
Enter the value to test: 3
3 is odd
~/projects/cpptest
>
```

Practical - 6 : Write a program to find the sum of natural numbers.

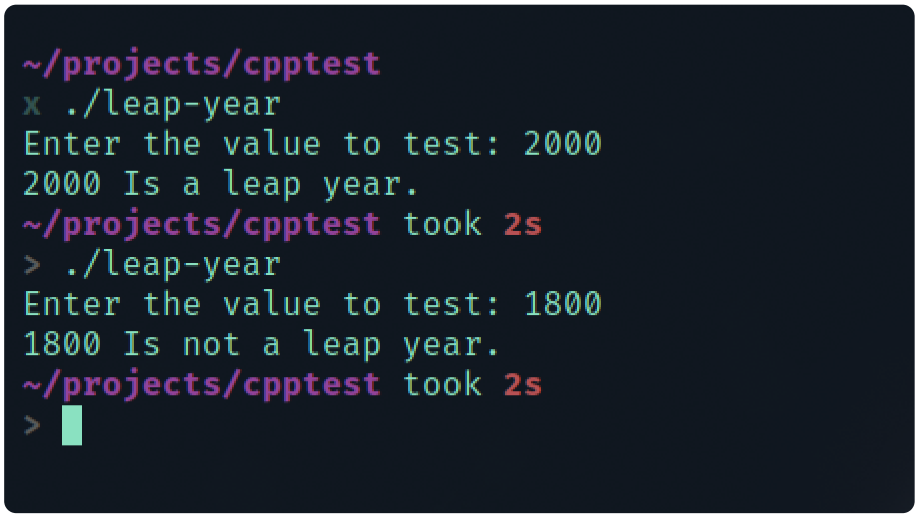
```
#include <iostream>
using namespace std;

class NaturalSum {
public:
    int naturalSumFn(int n) {
        return n * (n + 1) / 2;
    }
};

int main() {
    int n = 5;
    NaturalSum obj;
    cout << "Sum of first " << n << " natural numbers: " <<
obj.naturalSumFn(n);
    return 0;
}
```



```
~/projects/cpptest
> ./sum-natural
Sum of first 5 natural numbers: 15
~/projects/cpptest
>
```

Practical - 7 : Write a program to find leap year.

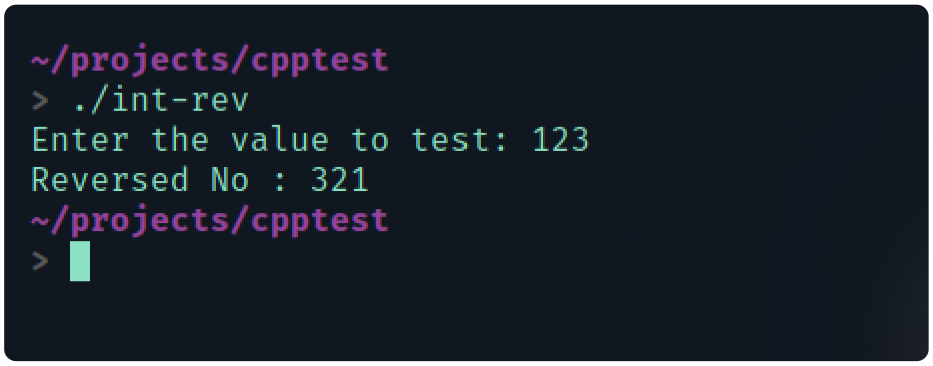
```cpp
#include <iostream>
using namespace std;

class Operations {
private:
        int year;

public:
        void isLeapYear(int year){
                if((year % 400 == 0) || ((year % 100 !=0) && (year %
4 == 0) )){
                        cout<<year<<" Is a leap year.";
                } else{
                        cout<<year<<" Is not a leap year.";
                }
        }
};

int main (){
        int a;
        Operations o1;
        cout<<"Enter the value to test: ";
        cin>>a;
        o1.isLeapYear(a);

        return 0;
}
```



```
~/projects/cpptest
x ./leap-year
Enter the value to test: 2000
2000 Is a leap year.
~/projects/cpptest took 2s
> ./leap-year
Enter the value to test: 1800
1800 Is not a leap year.
~/projects/cpptest took 2s
>
```

Practical – 8 : Write a program to reverse an integer.

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

class Operations {
private:
        int num, rev, reminder;

public:
        void reversed(int num){
                int reminder, rev = 0;

                while(num!=0){

                reminder = num % 10;
                rev = rev * 10 + reminder;
                num /= 10;
                }
        cout<<"Reversed No : "<<rev;
        }
};

int main (){
        int a;
        Operations o1;
        cout<<"Enter the value to test: ";
        cin>>a;
        o1.reversed(a);

        return 0;
}
```

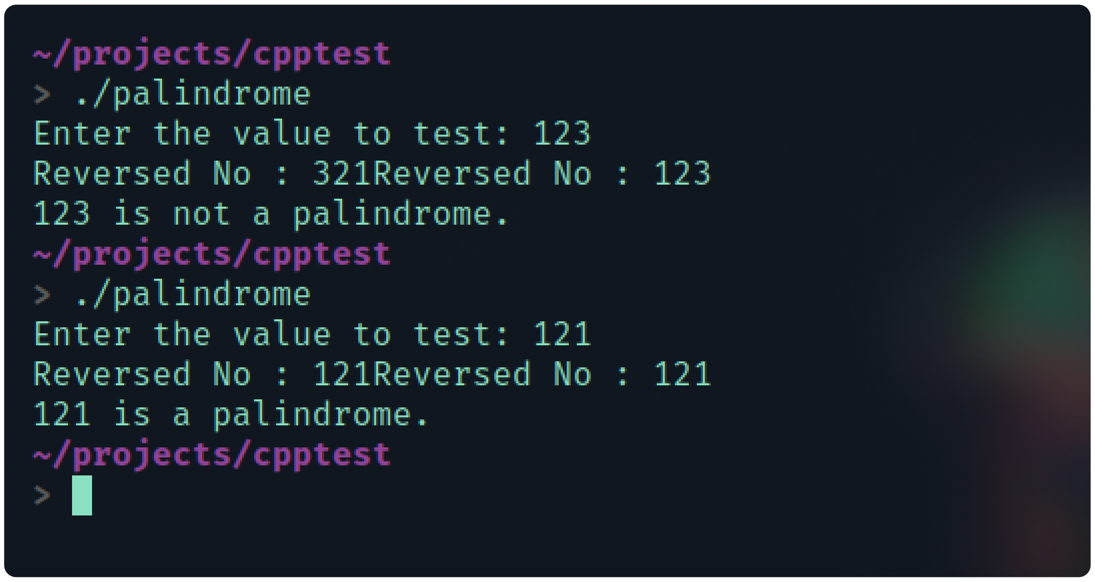


```
~/projects/cpptest
> ./int-rev
Enter the value to test: 123
Reversed No : 321
~/projects/cpptest
> 
```

Practical - 9 : Write a program to check whether the given integer is palindrome or not.

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

class Operations {
public:
        int reversed(int num){
                int reminder, rev = 0;

                while(num!=0){

                reminder = num % 10;
                rev = rev * 10 + reminder;
                num /= 10;
                }
        cout<<"Reversed No : "<<rev;
        return rev;
        }
};
```

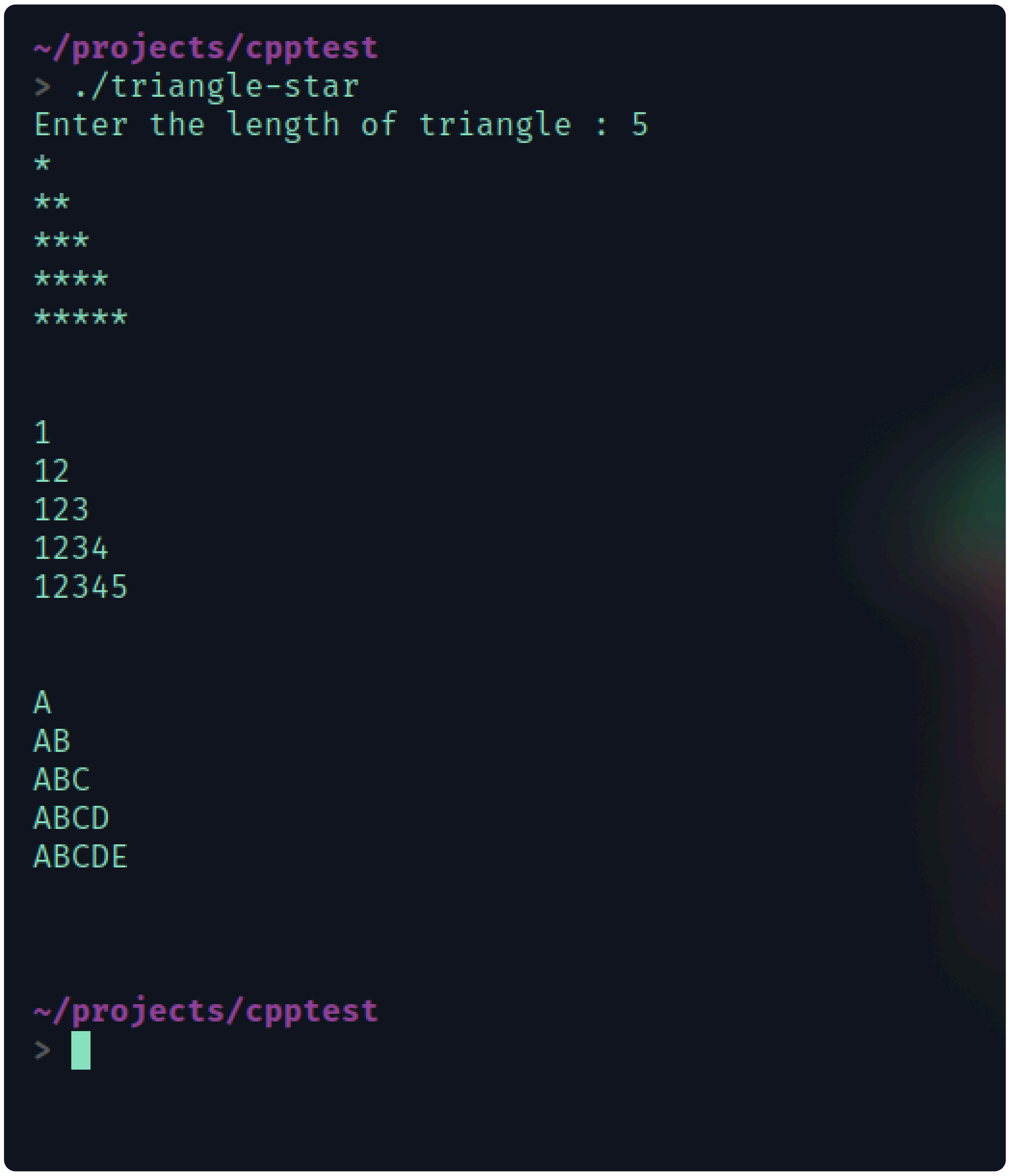
```cpp
int main (){
        int a;

        cout<<"Enter the value to test: ";
        cin>>a;

        Operations o1;
        a = o1.reversed(a);
        b = o1.reversed(a);
        if(a == b){
                cout<<a<<" is a palindrome."
        }else{
        cout<<a<<" is not a palindrome."
        }
        return 0;
}
```



```
~/projects/cpptest
> ./palindrome
Enter the value to test: 123
Reversed No : 321Reversed No : 123
123 is not a palindrome.
~/projects/cpptest
> ./palindrome
Enter the value to test: 121
Reversed No : 121Reversed No : 121
121 is a palindrome.
~/projects/cpptest
> 
```

Practical - 10 : Write a program that prints triangles using * , Numbers, Charecters .

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

class Operations {
public:
void triangleStar(int n){
        for(int i=1; i<=n; i++){
                for(int j=1; j<=i; j++){
                        cout<<"*";
                }
                cout<<endl;
        }
}

void triangleNum(int n){

        for(int i=1; i<=n; i++){
                for(int j=1; j<=i; j++){
                        cout<<j;
                }
                cout<<endl;
        }
}

void triangleChar(int n){

        for(int i=1; i<=n; i++){
                for(int j=1; j<=i; j++){
                        cout<<char('A' + j-1);
                }
                cout<<endl;
        }
}
};

int main (){
        int n;

        cout<<"Enter the length of triangle : ";
        cin>>n;

        Operations o1;
        o1.triangleStarr(n);
        o1.triangleNum(n);
        o1.triangleChar(n);
        return 0;

}
```

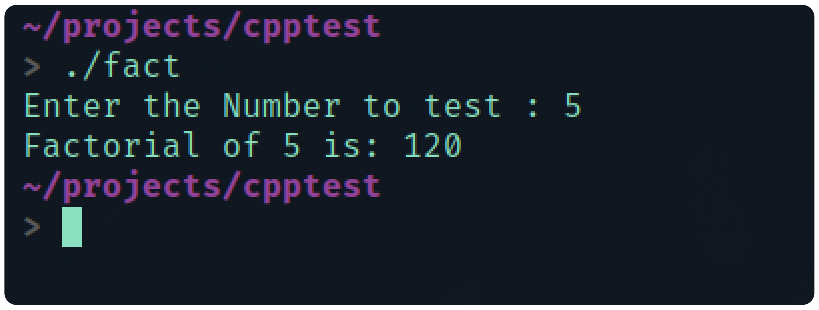


```
~/projects/cpptest
> ./triangle-star
Enter the length of triangle : 5
*
**
***
****
*****


1
12
123
1234
12345


A
AB
ABC
ABCD
ABCDE



~/projects/cpptest
> 
```

Practical - 11 : Write A Program to find the factorial of a given number.

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

class Factorial {
public:
```

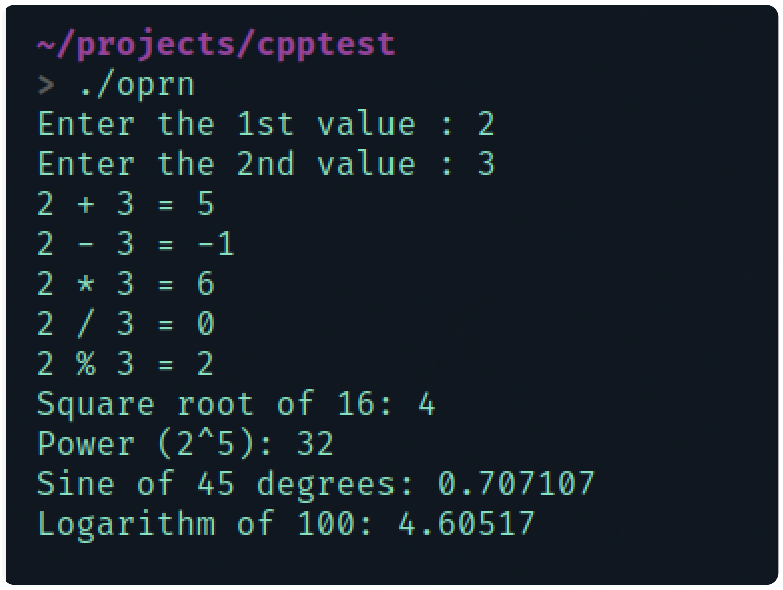
```cpp
    int findFactorial(int n) {
        int fact = 1;
        for (int i = 1; i <= n; ++i) {
            fact *= i;
        }
        return fact;
    }
};

int main() {
        int n;
        cout<<"Enter the Number to test : ";
        cin>>n;

    Factorial f1;
    cout << "Factorial of " << n << " is: " << f1.findFactorial(n);
    return 0;
}
```



```
~/projects/cpptest
> ./fact
Enter the Number to test : 5
Factorial of 5 is: 120
~/projects/cpptest
> 
```

Practical - 12 : Write a program to perform Mathematical Operations.

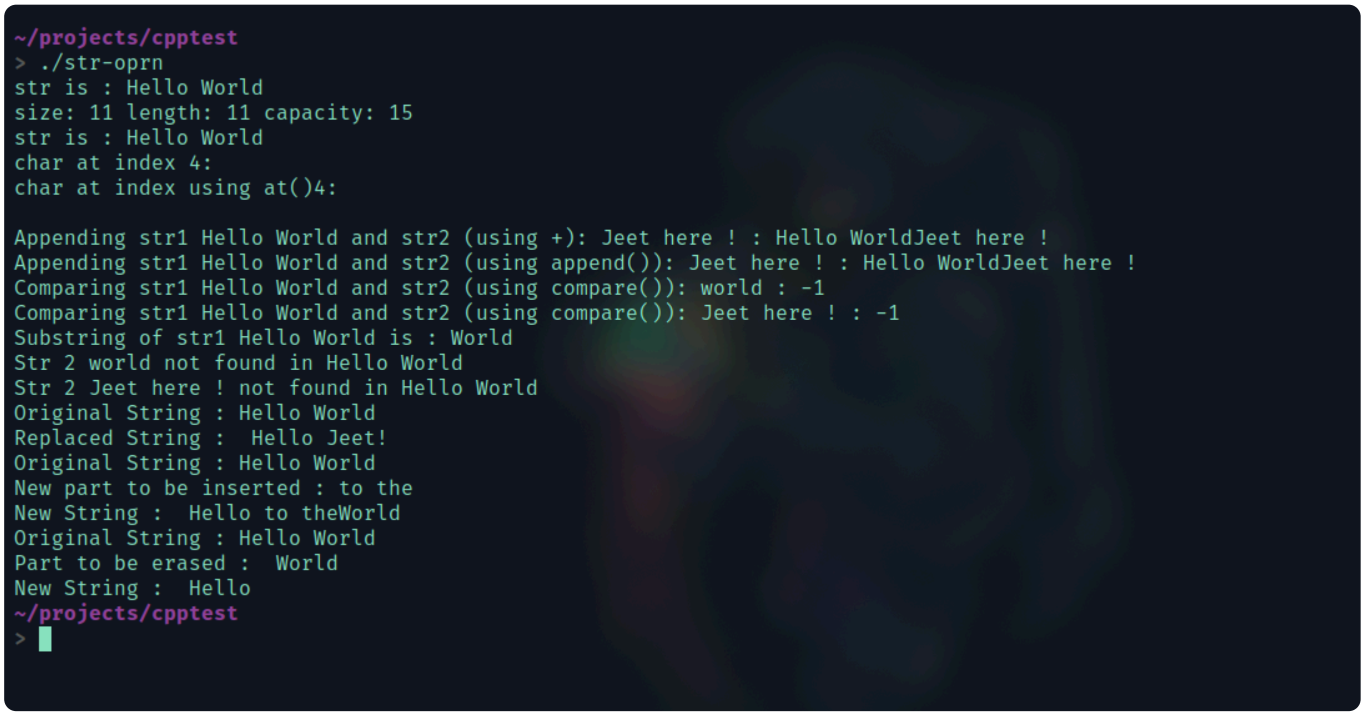
```cpp
#include <iostream>
#include <cmath>
using namespace std;

class Operations {
public:

void performOperations(int a, int b){
        cout<<a<<" + "<<b<<" = "<<a+b<<endl;
        cout<<a<<" - "<<b<<" = "<<a-b<<endl;
        cout<<a<<" * "<<b<<" = "<<a*b<<endl;
        cout<<a<<" / "<<b<<" = "<<a/b<<endl;
        cout<<a<<" % "<<b<<" = "<<a%b<<endl;
        cout << "Square root of 16: " << sqrt(16) << endl;
        cout << "Power (2^5): " << pow(2, 5) << endl;
        cout << "Sine of 45 degrees: " << sin(45 * M_PI / 180) <<
endl;
        cout << "Logarithm of 100: " << log(100) << endl;

}
};

int main (){
        int a,b,ch;
        Operations o1;

        cout<<"Enter the 1st value : ";
        cin>>a;

        cout<<"Enter the 2nd value : ";
        cin>>b;

        o1.performOperations(a,b);

        return 0;

}
```


```
~/projects/cpptest
> ./oprn
Enter the 1st value : 2
Enter the 2nd value : 3
2 + 3 = 5
2 - 3 = -1
2 * 3 = 6
2 / 3 = 0
2 % 3 = 2
Square root of 16: 4
Power (2^5): 32
Sine of 45 degrees: 0.707107
Logarithm of 100: 4.60517
```

Practical – 13 : Write a program to perform String Operations.

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

class Operations {
public:

void len(string s){
        cout << "str is : " << s << "\n";
        cout << "size: " << s.size() << " length: " << s.length() <<
" capacity: " << s.capacity() << "\n";
}

void access(string s, int a){
        cout << "str is : " << s << "\n";
        cout << "char at index "<<a<<" is : "s[a] << "\n";
        cout << "char at index using at()"<<a<<" is : "s.at(a) <<
"\n";
}

void apnd(string s1, string s2){
        cout<<"Appending str1 "<<s1<<" and str2 (using +): "<<s2<<" :
"<<s1+s2;
```

```cpp
cout<<"Appending str1 "<<s1<<" and str2 (using append()): "<<s2<<" :
"<<s1.append(s2);
}

void cmp(string s1, string s2){
        cout<<"Comparing str1 "<<s1<<" and str2 (using ==): "<<s2<<"
: "<<s1==s2;

cout<<"Comparing str1 "<<s1<<" and str2 (using compare()): "<<s2<<" :
"<<s1.compare(s2);
}

void substring(string s1, int index, int len){
        cout<<"Substring of str1 "<<s1<<" is : "<<s1.substr(index,
len);
}

void search(string s1, string s2){
        int res = s1.find(s2);
        if(res != string.npos){
                cout<<"Str 2 "<<s2<<" found at : "<<res;
        }else{
                cout<<"Str 2 "<< s2 <<" not found in "<<s1;
        }
}


void replace(string s1, int pos, int n, string s2){
        cout<<"Original String : "<<s1;
        cout<<"Replaced String :  "<<s1.replace(n, s2);
}

void insert(string s1, int pos, string s2){
        cout<<"Original String : "<<s1;
        cout<<"New part to be inserted : "<<s2;
        cout<<"New String :  "<<s1.insert(n, s2);
}

void erase(string s1, int pos){
        cout<<"Original String : "<<s1;
        cout<<"Part to be erased : "<<s1.substring(5,s1.length()-5);
        cout<<"New String :  "<<s1.erase(n);
}
};

int main (){
        string s1 = 'Hello World';
        string s2 = "Jeet here !";
        Operations o1;

        o1.len(s1);
        o1.access(s1,4);
        o1.append(s1,s2);
        o1.cmp(s1,"world");
        o1.cmp(s1,s2);
        o1.substr(s1,6,5);
        o1.search(s1,"world");
        o1.search(s1,s2);
        o1.replace(s1, 6, 5, "Jeet!");
        o1.insert(s1, 6, "to the");
        o1.erase(s1, 5);
return 0;

}
```



```
~/projects/cpptest
> ./str-oprn
str is : Hello World
size: 11 length: 11 capacity: 15
str is : Hello World
char at index 4:
char at index using at()4:

Appending str1 Hello World and str2 (using +): Jeet here ! : Hello WorldJeet here !
Appending str1 Hello World and str2 (using append()): Jeet here ! : Hello WorldJeet here !
Comparing str1 Hello World and str2 (using compare()): world : -1
Comparing str1 Hello World and str2 (using compare()): Jeet here ! : -1
Substring of str1 Hello World is : World
Str 2 world not found in Hello World
Str 2 Jeet here ! not found in Hello World
Original String : Hello World
Replaced String :  Hello Jeet!
Original String : Hello World
New part to be inserted : to the
New String :  Hello to theWorld
Original String : Hello World
Part to be erased :  World
New String :  Hello
~/projects/cpptest
> 
```

Practical - 14 : Write A Program to create a Book class and access data members through object.

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

class Book {

private:
        int ISBN;
        string bookName;
        string authorName;
        double price;

public:

Book () : ISBN(0), bookName(""), authorName(""), price(0){
        // Default paramters
}

Book(int isbn, string name, string author, double amt){
        ISBN = isbn;
        bookName = name;
        authorName = author;
        price = amt;
}

int getISBN(){
        return ISBN;
}

string getBookName(){
        return bookName;
}

string getAuthorName(){
        return authorName;
}
```

```cpp
double getPrice(){
        return price;
}

void displayBookInfo(){
        cout<<"ISBN Number : "<<ISBN<<endl;
        cout<<"Book Title : "<<bookName<<endl;
        cout<<"Author Name : "<<authorName<<endl;
        cout<<"Price : "<<price<<endl;
}
};

int main(){

Book book[5];
book[0] = Book();
book[1] = Book(1001, "Think and Grow Rich", "Napolean Hill", 1499);
book[2] = Book(1002, "As a man Thinketh", "James Allen", 2999);
book[3] = Book(1003, "Meditations", "Marcus Aurellius", 1999);
book[4] = Book(1004, "Sherlock Holmes", "Arthur Conan Doyale", 1499);


for(int i=0; i<=5; i++){

cout<<"Book no: "<<i<<endl;
cout<<"ISBN : "<<book[i].getISBN()<<endl;
cout<<"Title : "<<book[i].getBookName()<<endl;
cout<<"Author : "<<book[i].getAuthorName()<<endl;
cout<<"Price : "<<book[i].getPrice()<<endl;
book[i].displayBookInfo();

cout<<endl<<endl;
}

return 0;
}
```

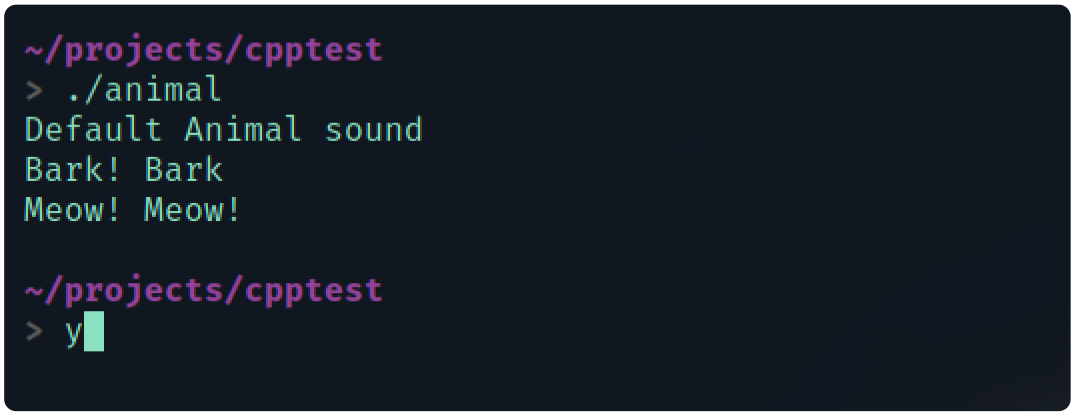
```
Book no: 1
ISBN : 1001
Title : Think and Grow Rich
Author : Napolean Hill
Price : 1499
ISBN Number : 1001
Book Title : Think and Grow Rich
Author Name : Napolean Hill
Price : 1499


Book no: 2
ISBN : 1002
Title : As a man Thinketh
Author : James Allen
Price : 2999
ISBN Number : 1002
Book Title : As a man Thinketh
Author Name : James Allen
Price : 2999


Book no: 3
ISBN : 1003
Title : Meditations
Author : Marcus Aurellius
Price : 1999
ISBN Number : 1003
Book Title : Meditations
Author Name : Marcus Aurellius
Price : 1999
```

Practical – 15 : Write a program to create an Animal Class (use Inheritance) :

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

class Animal {
private:

        string name;
        int legs;
        string sound;
```

```cpp
public:

Animal() : name(""), sound(""), legs(0){
        // Default Constructor
}

Animal(string animalName, string animalSound, int animalLegs){
        name = animalName;
        sound = animalSound;
        legs = animalLegs;
}

void speak(){
        cout<<"Default Animal sound\n";
}
};

class Dog:Animal{
public:

void speak(){
        cout<<"Bark! Bark\n";
}
};

class Cat:Animal{
public:

void speak(){
        cout<<"Meow! Meow!\n";
}
};

int main(){

Animal a1;
a1.speak();

Dog G;
G.speak();
```

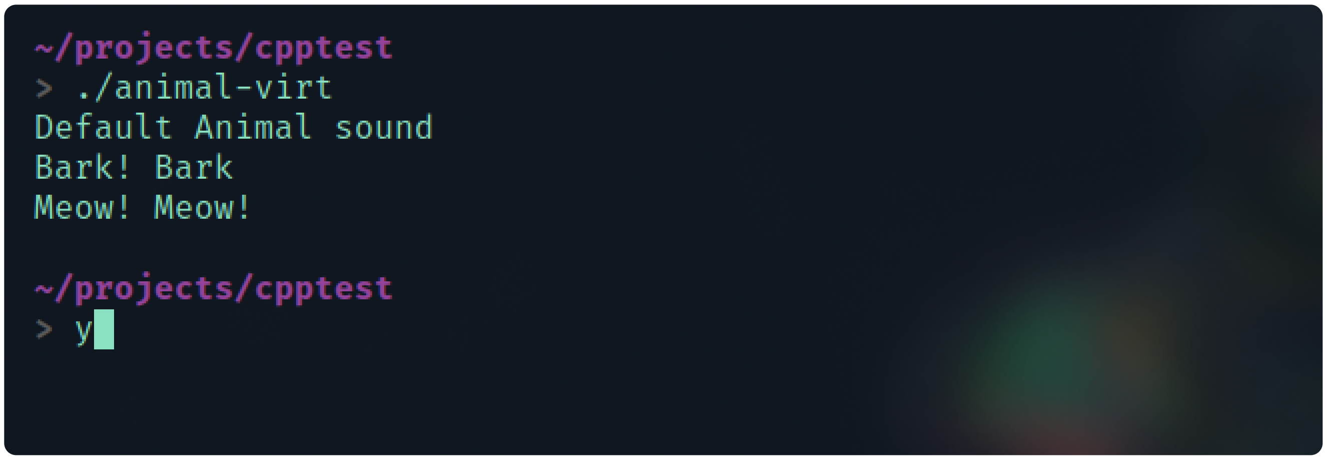
```
Cat Bob;
Bob.speak();

return 0;
}
```



```
~/projects/cpptest
> ./animal
Default Animal sound
Bark! Bark
Meow! Meow!

~/projects/cpptest
> y
```

Practical – 16 : Write A Program using method override and virtual keyword in class.

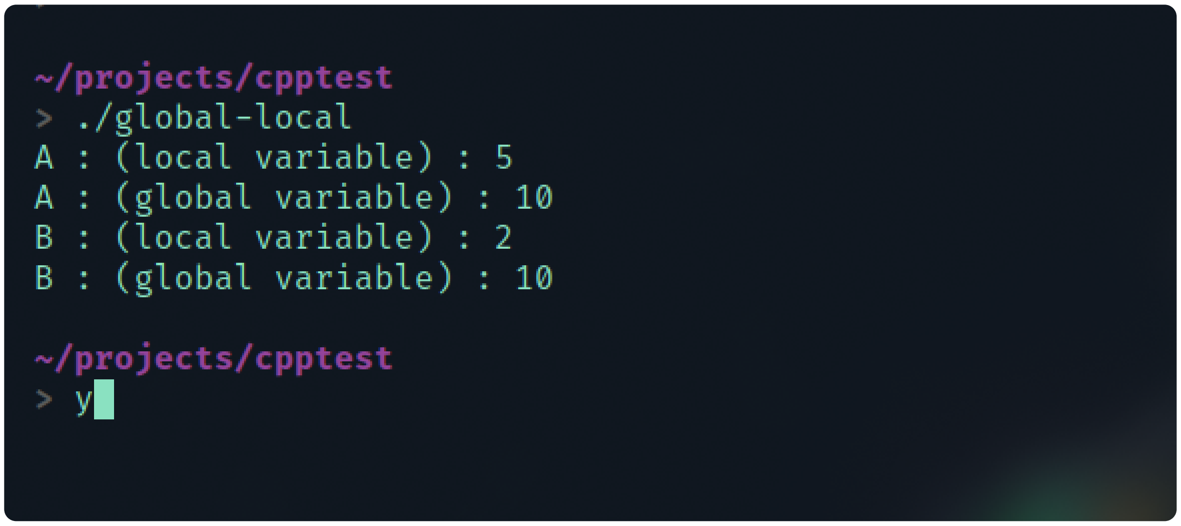
```
#include <iostream>
using namespace std;

class Animal {
private:

        string name;
        int legs;
        string sound;

public:

Animal() : name(""), sound(""), legs(0){
        // Default Constructor
}

Animal(string animalName, string animalSound, int animalLegs){
        name = animalName;
        sound = animalSound;
        legs = animalLegs;
}

virtual void speak(){
        cout<<"Default Animal sound\n";
}
};

class Dog:Animal{
public:

void speak() override{
        cout<<"Bark! Bark\n";
}
};

class Cat:Animal{
public:

void speak() override{
        cout<<"Meow! Meow!\n";
}
};

int main(){

Animal a1;
a1.speak();

Dog G;
G.speak();

Cat Bob;
Bob.speak();

return 0;
}
```



```
~/projects/cpptest
> ./animal-virt
Default Animal sound
Bark! Bark
Meow! Meow!

~/projects/cpptest
> y
```

Practical - 17 : Write A Program to demonstrate how to access local variables and global variables.

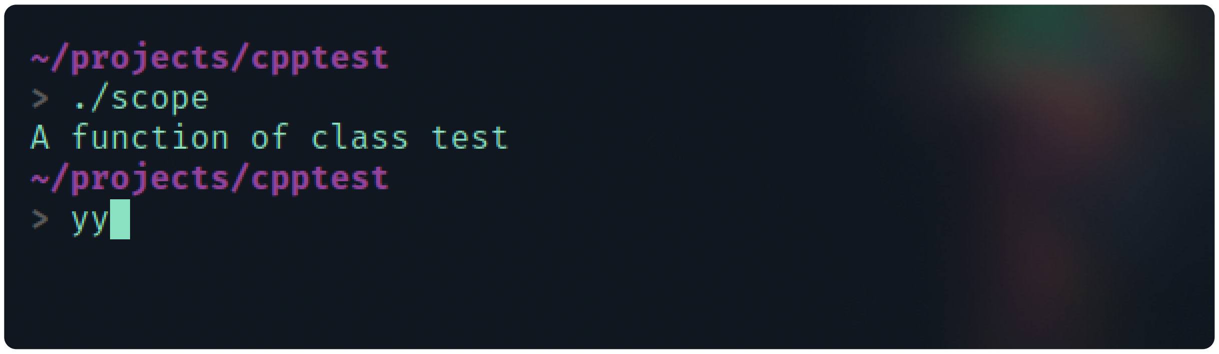
```cpp
#include <iostream>
using namespace std;

int a = 10, b = 10;

class Test{
public:

void scope(){
        int a = 5;

        cout<<"A : (local variable) : "<< a << endl;
        cout<<"A : (global variable) : "<< ::a << endl;
}
};

int main(){
        int b = 2;

        Test t1;
        t1.scope();

        cout<<"B : (local variable) : "<< b << endl;
        cout<<"B : (global variable) : "<< ::b << endl;
        return 0;
}
```



```
~/projects/cpptest
> ./global-local
A : (local variable) : 5
A : (global variable) : 10
B : (local variable) : 2
B : (global variable) : 10

~/projects/cpptest
> y
```

Practical – 18 : Write A Program to use of Scope resolution operator (method defined outside class).

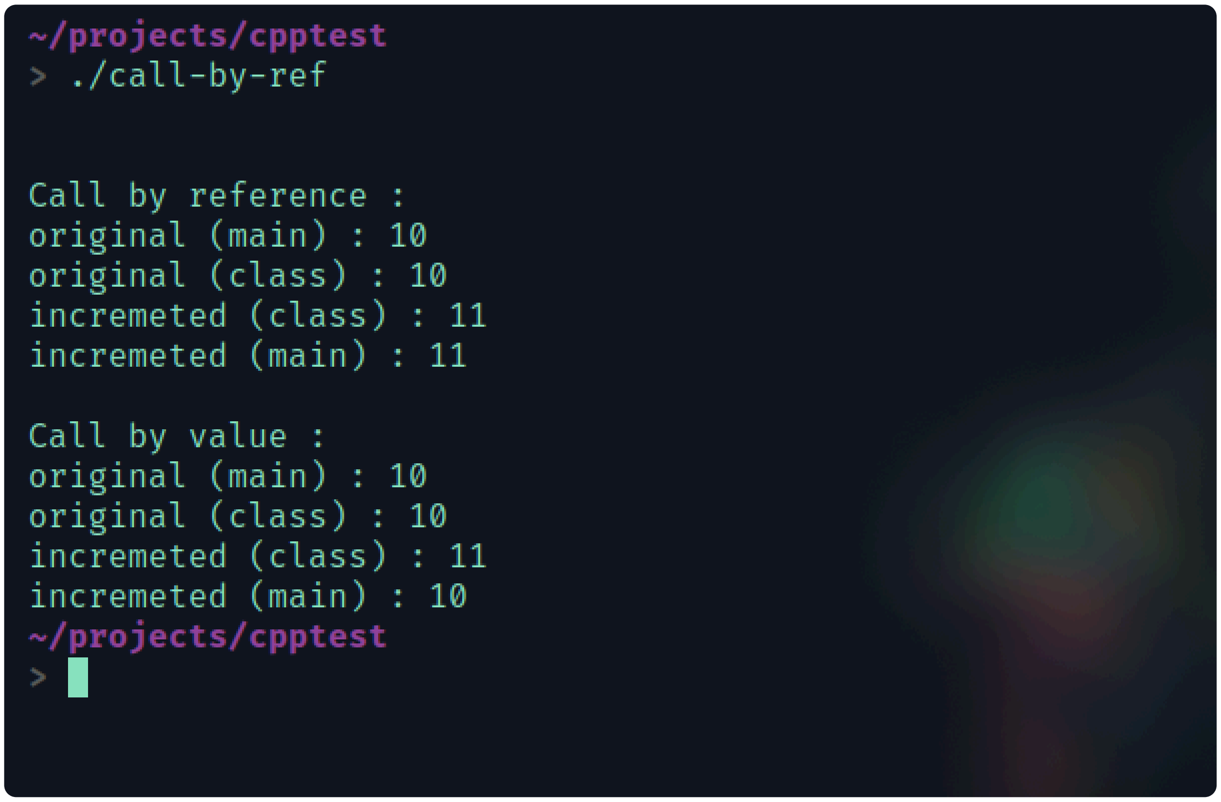
```
#include <iostream>
using namespace std;

class Test{

public:
        void disp();
};

void Test:: disp(){
        cout<<"A function of class test";
}

int main(){
        Test t1;
        t1.disp();
        return 0;
}
```



```
~/projects/cpptest
> ./scope
A function of class test
~/projects/cpptest
> yy
```

Practical - 19 : Write A Program to call by value and call by reference.

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

class Test{
public:

void ref(int &a){
        cout<<"\noriginal (class) : "<<a;
        a++;
        cout<<"\nincremeted (class) : "<<a;
}

void val(int b){
        cout<<"\noriginal (class) : "<<b;
        b++;
        cout<<"\nincremeted (class) : "<<b;
}
};

int main(){
int a = 10, b = 10;

cout<<"\n\nCall by reference : ";
cout<<"\noriginal (main) : "<<a;
Test t1;
t1.ref(a);
cout<<"\nincremeted (main) : "<<a<<" (value changed)";

cout<<"\n\nCall by value : ";
cout<<"\noriginal (main) : "<<b;
t1.val(b);
cout<<"\nincremeted (main) : "<<b;
return 0;
}
```

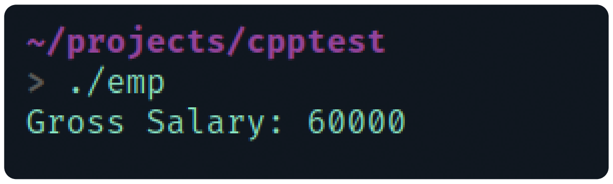


```
~/projects/cpptest
> ./call-by-ref


Call by reference :
original (main) : 10
original (class) : 10
incremeted (class) : 11
incremeted (main) : 11

Call by value :
original (main) : 10
original (class) : 10
incremeted (class) : 11
incremeted (main) : 10
~/projects/cpptest
>
```

Practical - 20 : Write A Program to calculate gross salary of an Employee.

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

class Employee {
public:
    float basicSalary, DA, HRA, grossSalary;

    void grossSalaryCalc() {
```

```cpp
        DA = 0.8 * basicSalary;
        HRA = 0.2 * basicSalary;
        grossSalary = basicSalary + DA + HRA;
    }

    void displayGrossSalary() {
        cout << "Gross Salary: " << grossSalary << endl;
    }
};

int main() {
    Employee emp;
    emp.basicSalary = 30000;
    emp.grossSalaryCalc();
    emp.displayGrossSalary();
    return 0;
}
```



```
~/projects/cpptest
> ./emp
Gross Salary: 60000
```

🚀 Github link for the Assignment

https://github.com/JeetChauhan17/OOP-Practical

📝 MADE BY `JEET CHAUHAN` (15742)