

Exercises chapter3

Lab 5 and 6

Type Cast

A Cast operator is a unary operator which forces one data type to be converted into another data type.

C++ supports four types of casting such as:

- Implicit conversion
- Explicit conversion
- Static Cast

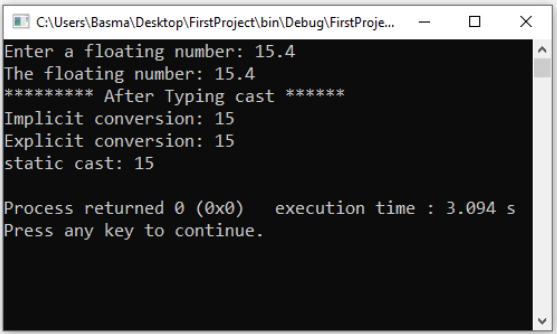
Exercise: Write a C++ program that takes a float number then convert it to integer with different ways.

```
#include <iostream>
using namespace std;
int main()
{
    float number;

    cout<<"Enter a floating number: ";
    cin>> number;
    cout<<"The floating number: "<< number<<endl;

    int way1 = number; // Implicit conversion
    int way2 = (int)number; // Explicit conversion
    int way3 = static_cast<int>(number); // static cast

    cout<<"***** After Typing cast *****"<<endl;
    cout <<"Implicit conversion: "<<way1<<endl;
    cout<<"Explicit conversion: "<< way2 <<endl;
    cout<<"static cast: "<< way3 <<endl;
    return 0;
}
```



The screenshot shows a terminal window with the following output:

```
C:\Users\Basma\Desktop\FirstProject\bin\Debug\FirstProje...
Enter a floating number: 15.4
The floating number: 15.4
***** After Typing cast *****
Implicit conversion: 15
Explicit conversion: 15
static cast: 15

Process returned 0 (0x0)   execution time : 3.094 s
Press any key to continue.
```

For more visit: <http://www.cplusplus.com/doc/tutorial/typecasting/>

Functions

A function is a group of statements that is executed when it is called from some point of the program.

1- Predefined Functions

C++ provides a library of predefined functions. The definitions of many common functions are found in the `cmath` and `cstdlib`. Functions are `pow` (parameter1, parameter2) (*like* x^2), `sqrt`(parameter1) (like \sqrt{x}), `abs`(parameter1), and `floor`(parameter1) such as `floor(48.79) = 48.0`

For more functions go to the following link: <http://www.cplusplus.com/reference/cmath/>

To access these functions, your program must include `cmath` and `cstdlib` library:

```
#include <cmath>
#include <cstdlib>
```

To call these functions:

```
Function_Name(Parameter1, Parameter2, ...)
```

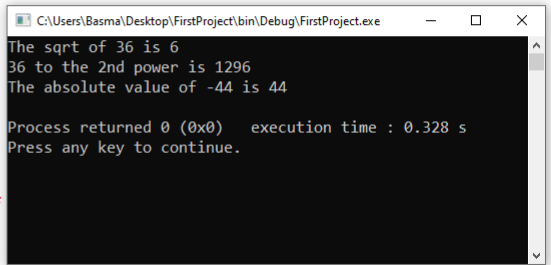
Exercises:

Exercise 1: Use `sqrt`, `pow`, and `abs` function with `cout` function.

```
#include <iostream>
#include <cmath>
#include <cstdlib>

using namespace std;

int main ( )
{
    cout << "The sqrt of 36 is " << sqrt( 36 ) << endl;
    cout << "36 to the 2nd power is " << pow( 36, 2 ) << endl;
    cout << "The absolute value of -44 is " << abs( -44.0 ) << endl;
    return 0;
}
```



Exercise 2: Write a C++ program that take a number from user then output the square of this number.

Enter a number: 4

The square: 2

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

int main ( )
{
    int number;

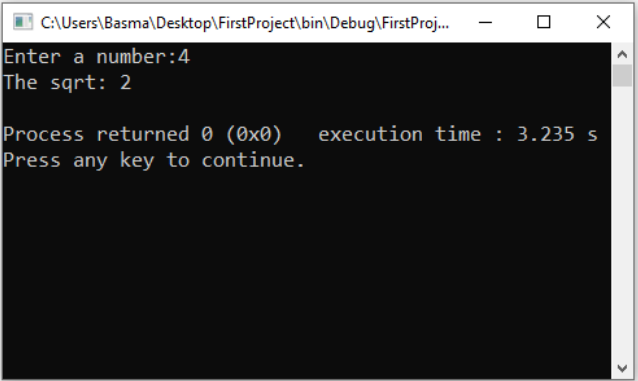
    cout<<"Enter a number:";

    cin>> number;

    int result = sqrt(number);

    cout << "The sqrt: " << result << endl;

    return 0;
}
```



2- Defined Functions

Syntax: type name (parameter1, parameter2, ...) { statements }

Where

- type is the data type specifier of the data returned by the function.
- name is the identifier by which it will be possible to call the function.
- parameters (as many as needed): Each parameter consists of a data type specifier followed by an identifier, like any regular variable declaration (for Exercise: int x) and which acts within the function as a regular local variable. They allow to pass arguments to the function when it is called. The different parameters are separated by commas.
- statements is the function's body. It is a block of statements surrounded by braces { }.

Types of User-defined Functions in C++

1- Function that doesn't take parameter and doesn't return result.

Exercise: Write a function that print a message: Welcome, this is a function.

```

1  #include <iostream>
2  using namespace std;
3
4  void fun1()
5  {
6      cout<<"Welcome, this is a function"<<endl;
7  }
8
9  int main()
10 {
11     fun1();
12 }

```

Output window: C:\Users\Basma\Desktop\FirstProject\bin\Debug\FirstProject.exe
Welcome, this is a function
Process returned 0 (0x0) execution time : 0.271 s
Press any key to continue.

2- Function that take a parameter and doesn't return

Exercise: Write a function that take your name and print a message say: Welcome, your name.

```

#include <iostream>
using namespace std;

void printName(string name)
{
    cout<<"Welcome, "<< name<<endl;
}

int main()
{
    string n;

    cout<<"Enter your name: ";
    cin>> n;

    printName(n);
}

```

Output window: Select C:\Users\Basma\Desktop\FirstProject\bin\Debug\FirstP...
Enter your name: Basma
Welcome, Basma
Process returned 0 (0x0) execution time : 3.153 s
Press any key to continue.

3- Function doesn't take an argument but return a value

Exercise: Write a function that return your name.

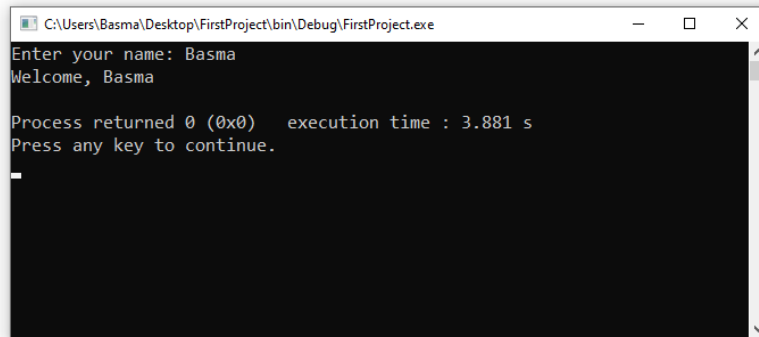
```
#include <iostream>
using namespace std;

string getName()
{
    string n;

    cout<<"Enter your name: ";
    cin>> n;

    return n;
}

int main()
{
    string n = getName();
    cout<<"Welcome, "<< n<<endl;
}
```



4- Function with argument and return value

Exercise: write a function that take a number and return it is type odd or even.

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

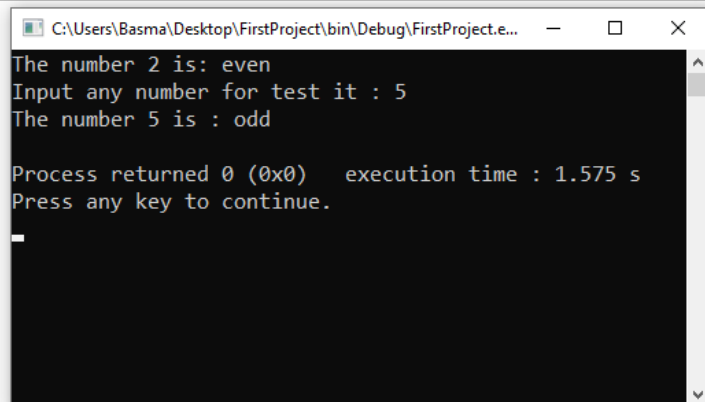
string odd_even(int number)
{
    string result;

    if (number%2 == 0)
        result= "even";
    else
        result= "odd";
    return result;
}

int main()
{
    cout<<"The number 2 is: "<< odd_even(2)<<endl;
    int num;
    string result;

    cout<<"Input any number for test it : ";
    cin>>num;

    result = odd_even(num);
    cout<<"The number "<< num<<" is : "<< result <<endl;
}
```



Difference between Local and Global Variable

Variables that are declared inside a function or a block are called local variables and are said to have local scope. These local variables can only be used within the function or block in which these are declared.

Variables that are defined outside of all the functions and are accessible throughout the program are global variables and are said to have global scope.

Parameter	Local	Global
Scope	It is declared inside a function.	It is declared outside the function.
Value	If it is not initialized, a garbage value is stored	If it is not initialized zero is stored as default.
Lifetime	It is created when the function starts execution and lost when the functions terminate.	It is created before the program's global execution starts and lost when the program terminates.
Data sharing	Data sharing is not possible as data of the local variable can be accessed by only one function.	Data sharing is possible as multiple functions can access the same global variable.
Modification of variable value	When the value of the local variable is modified in one function, the changes are not visible in another function.	When the value of the global variable is modified in one function changes are visible in the rest of the program.
Accessed by	Local variables can be accessed with the help of statements, inside a function in which they are declared.	You can access global variables by any statement in the program.
Memory storage	It is stored on the stack unless specified.	It is stored on a fixed location decided by the compiler.

Exercise:

```

#include <iostream>
using namespace std;

int g_variable;

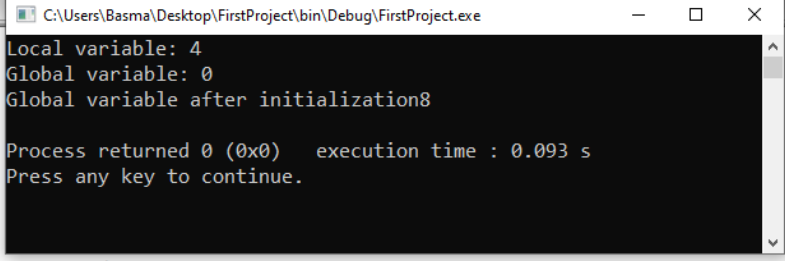
int main()
{
    int loc_variable = 4;

    cout << "Local variable: " <<loc_variable << endl;
    cout << "Global variable: " <<g_variable << endl;

    g_variable = loc_variable * 2;

    cout << "Global variable after initialization" <<g_variable << endl;
    return 0;
}

```



More Exercises

Exercise 1: Write a program in C to find the square of any number using the function.

```

#include <iostream>
using namespace std;

double square(double num)
{
    return (num * num);
}

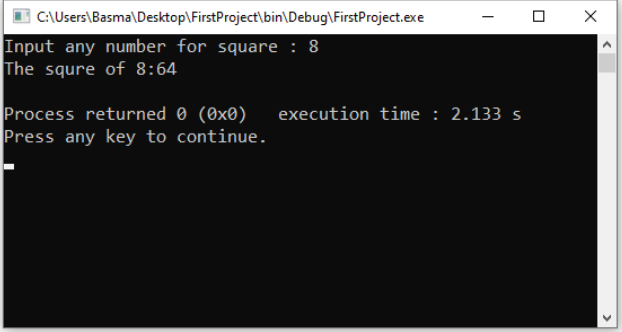
int main()
{
    int num;
    double result;

    cout<<"Input any number for square : ";
    cin>>num;

    result = square(num);

    cout<<"The square of "<< num<<': ' << result<<endl;
    return 0;
}

```



Exercise 2: Write a C++ function to find the Max of two numbers.

```

#include <iostream>
using namespace std;
int getMax(int num1, int num2)
{
    int the_max= 0;
    if (num1 > num2)
        the_max= num1;

    else
        the_max= num2;

    return the_max;
}
int main()
{
    int get_max = getMax(4,5);
    cout<<"The max of numbers 4 and 5 is:"<<get_max<<endl;
}

```

Exercises 3: Write a C++ program that will display the calculator menu.

The program will prompt the user to choose the operation choice (from 1 to 5). Then it asks the user to input two integer vales for the calculation. See the sample below.

MENU

1. Add
2. Subtract
3. Multiply
4. Divide
5. Modulus

Enter your choice: 1

Enter your two numbers: 12 15

Result: 27

```
#include <cstdlib>
```

```
#include <iostream>
```

```
using namespace std;
```

```
void displaymenu(){
```

```
cout<<"======"<<"\n";
```



```
cout<<"          MENU          "<<"\n";
cout<<"===== "<<"\n";
cout<<"  1.Add"<<"\n";
cout<<"  2.Subtract"<<"\n";
cout<<"  3.Multiply"<<"\n";
cout<<"  4.Divide"<<"\n";
cout<<"  5.Modulus"<<"\n";
    }
int Add(int a,int b){
    return(a+b);
}

int Substract(int a, int b){
    return(a-b);
}

int Multiply(int a, int b){
    return(a*b);
}

float Divide(int a,int b){
    return(a/b);
}

int Modulus(int a, int b){
    return(a%b);
}

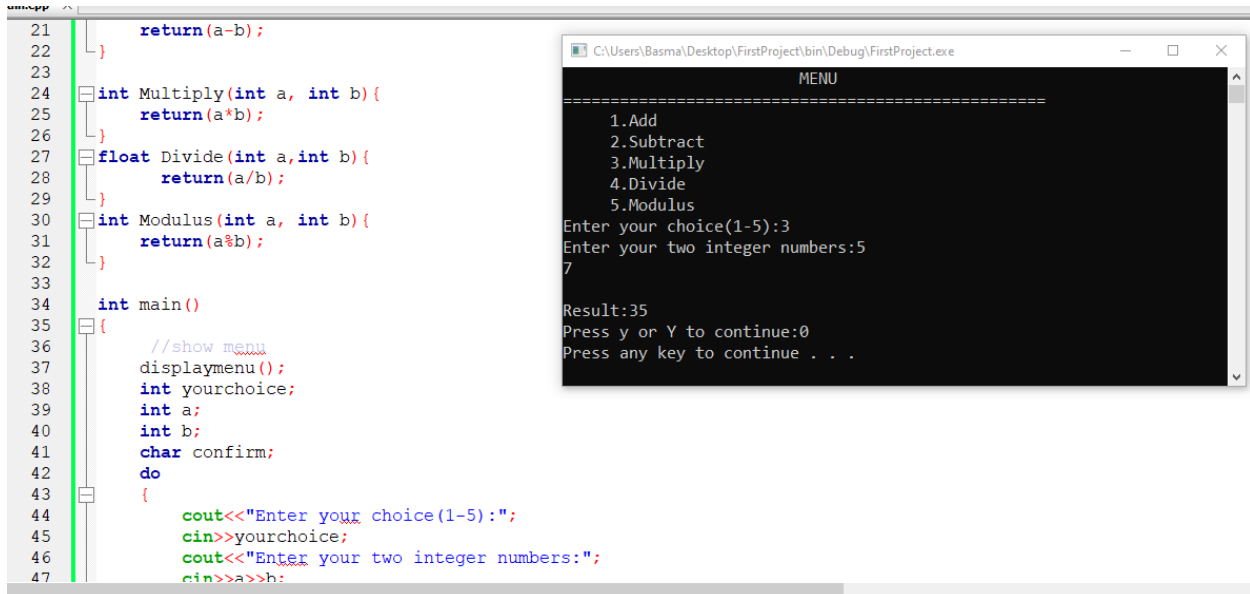
int main()
{
```

```
//show menu
displaymenu();
int yourchoice;
int a;
int b;
char confirm;
do
{
    cout<<"Enter your choice(1-5):";
    cin>>yourchoice;
    cout<<"Enter your two integer numbers:";
    cin>>a>>b;
    cout<<"\n";
    switch(yourchoice)
    {
        case 1:cout<<"Result:"<<Add(a,b);break;
        case 2:cout<<"Result:"<<Substract(a,b);break;
        case 3:cout<<"Result:"<<Multiply(a,b);break;
        case 4:cout<<"Result:"<<Divide(a,b);break;
        case 5:cout<<"Result:"<<Modulus(a,b);break;
        default:cout<<"invalid";
    }

    cout<<"\nPress y or Y to continue:";
    cin>>confirm;
}

while(confirm=='y'||confirm=='Y');
```

```
system("PAUSE");  
return EXIT_SUCCESS;  
}
```



The screenshot shows a C++ IDE with a code editor on the left and a terminal window on the right. The code in the editor includes functions for addition, subtraction, multiplication, division, and modulus, along with a main function that displays a menu and takes user input. The terminal window shows the output of the program, including the menu, user choices, and the resulting calculations.

```
21     return(a-b);  
22 }  
23  
24 int Multiply(int a, int b){  
25     return(a*b);  
26 }  
27 float Divide(int a,int b){  
28     return(a/b);  
29 }  
30 int Modulus(int a, int b){  
31     return(a%b);  
32 }  
33  
34 int main()  
35 {  
36     //show menu  
37     displaymenu();  
38     int yourchoice;  
39     int a;  
40     int b;  
41     char confirm;  
42     do  
43     {  
44         cout<<"Enter your choice(1-5):";  
45         cin>>yourchoice;  
46         cout<<"Enter your two integer numbers:";  
47         cin>>a>>b;
```

Terminal Output:

```
-----  
MENU  
-----  
1.Add  
2.Subtract  
3.Multiply  
4.Divide  
5.Modulus  
Enter your choice(1-5):3  
Enter your two integer numbers:5  
7  
Result:35  
Press y or Y to continue:0  
Press any key to continue . . .
```

Homework tasks

Task 1: Write a C++ program that take a number from user then output the power of this number.

Enter a number: 4

The power: 1

Task 2: Write a function take three numbers from user then output the minimum number.

Task 3: Write function that take a string then reverse it.

Sample String: "1234abcd"

Expected Output: "dcba4321"