# Lab #1

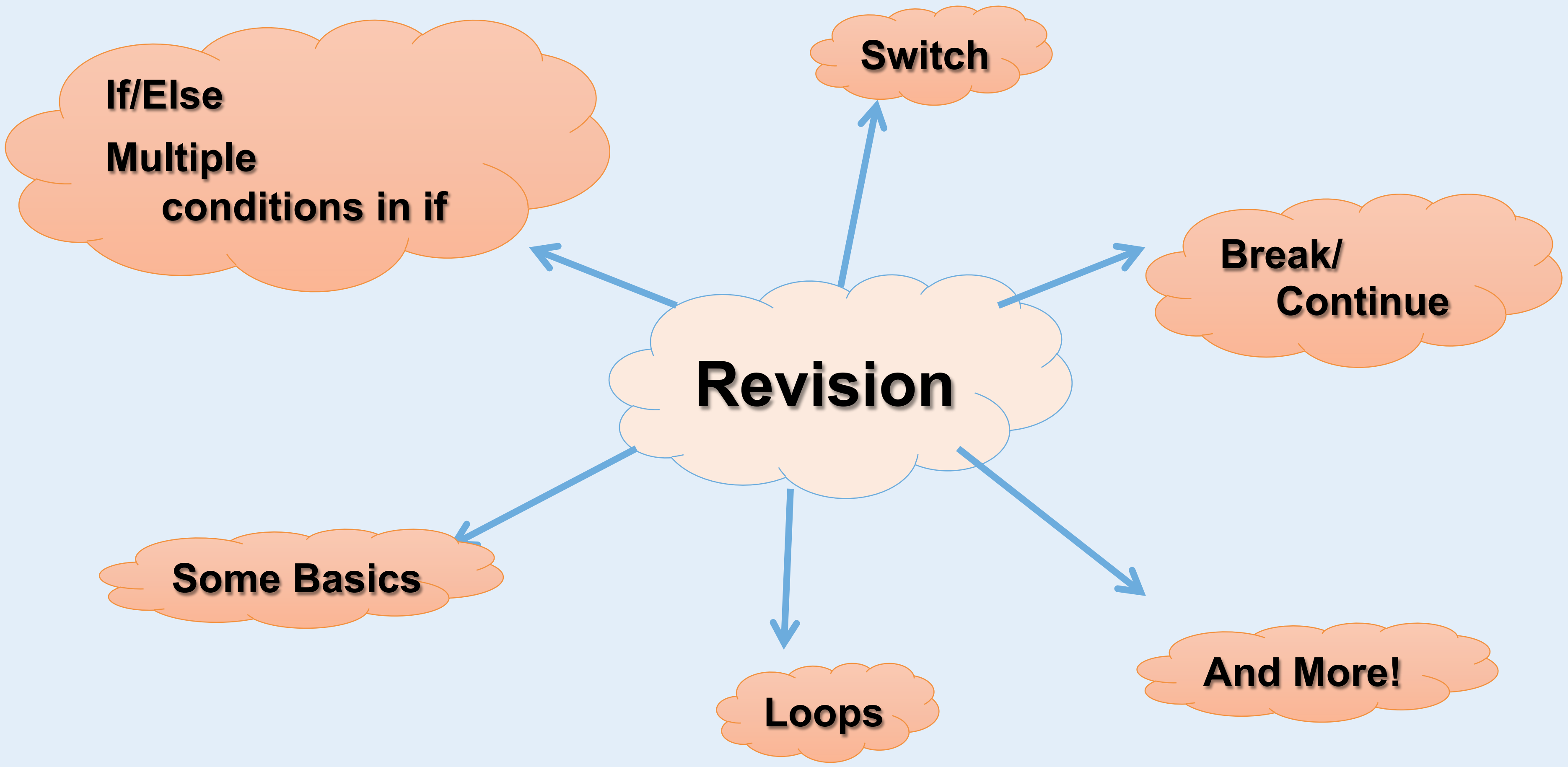
# Basics + Flow Control

**Structured Programming 2017/2018**

## Lab Rules

1. You MUST attend the lecture. The section is for practical work ONLY.
2. Attendance limited to student section. No EXCEPTIONS.
3. Attendance will be taken through QR code.
4. Lab Content will be sent weekly your dropbox folder.
5. Assignment degrees will be granted on lab tasks
6. You must build a team for the SP projects (Min 4 members, Max 6 members)
7. Be ready for QUIZ at anytime.

# Today's Lab


Revision
If/Else
Multiple conditions in if
Switch
Break/ Continue
Some Basics
Loops
And More!

## Tracing (1)

What is wrong in the following code?

```
#include <iostream>
using namespace std;
int main()
{
    int a, b, f=0;

    cout << "Enter a number to get its factorial: ";
    cin >> b;
    while (b != 0)
    {
           f*=b;
           b--;
    }
    cout << "The factorial =: " <<f<<endl;
}
```

f should be initialized to 1

f always equals 0

## Tracing (2)

When does the following program output the word ***excellent***?

What happens if we eliminate the ***else*** keyword?

```
#include <iostream>
using namespace std;
int main()
{
        float grade;
        cin>>grade;
        if(grade > 75)
                cout<<"very good\n";
        ~~else~~ if(grade > 85)
                cout<<"excellent\n";
}
```

“Exce

It'll output very good then excellent for the grades above 85

## Tracing (2) – cont'd

To correct the program, we should swap the ***if*** and the ***else*** statements:

```
#include <iostream>
using namespace std;
int main()
{
        float grade;
        cin>>grade;
        if(grade > 85)
                cout<<"excellent\n";
        else if(grade > 75)
                cout<<"very good\n";
}
```

## Tracing (3)

When does the following program output the word ***no***?

```
#include <iostream>
using namespace std;
int main()
{
        int x = 1;
        if ( x = 4 )
                cout << "yes";
        else
                cout << "no";
}
```

Never, the operator used here is <u>assignment</u> not <u>comparison</u> operator '=='

# Tracing (4)

What happens if we enter ***2 then 3 then 3*** in the following program?

a. Would the ***else*** be invoked?

```
int a,b,c;
cout <<"enter 3 numbers a, b, a
cin>> a >> b >> c ;
if ( a == b )
      if ( b == c )
            cout <<"a, b and c are the same
else
      cout << "a and b are different \n";
```

**The else will not be invoked.**
**The else is matched with the last if**

## Tracing (4) – cont'd

**b.** Correct the program such that the ***else*** matches the second ***if***:

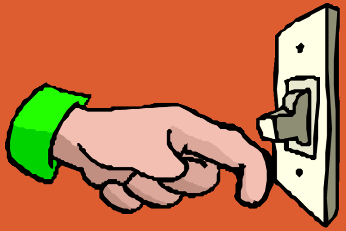
```
int a,b,c;
cout <<"enter 3 numbers a, b, and c \n";
cin>> a >> b >> c ;
if ( a == b )
        if ( b == c )
                cout <<"a, b and c are the same \n";
        else
                cout << “b and c are different \n";
```

## Tracing (4) – cont’d

c. Correct the program such that the ***else*** matches the first ***if***:

```
int a,b,c;
cout <<"enter 3 numbers a, b, and c \n";
cin>> a >> b >> c ;
if ( a == b )
{
        if ( b == c )
                cout <<"a, b and c are the same \n";
}

else
        cout << “a and b are different \n";
```

# Tracing (5)

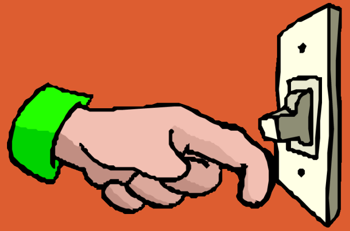


a. What is the output if we entered letter ‘***B***’?

```
char letter;
cout <<"enter a letter\n" ;
cin>> letter ;
switch(letter)
{
    case 'A': cout<<"you typed A\n";
    case 'B': cout<<"you typed B\n";
    default:  cout<<"Invalid input\n";
}
```

You typed B.
Invalid Input

## Tracing (5) – cont'd

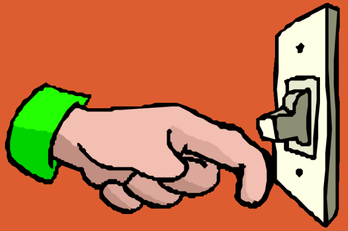


**b.** How to fix this bug?

Add **break** after each **case**

```
char letter;
cout <<"enter a letter\n" ;
cin>> letter ;
switch(letter)
{
    case 'A': cout<<"you typed A\n"; break;
    case 'B': cout<<"you typed B\n"; break;
    default:  cout<<"Invalid input\n";
}
```

## Tracing (5) – cont'd



c. Modify the code to handle input of small letters '***a***' and '***b***':

```
char letter;
cout <<"enter a letter\n" ;
cin>> letter ;
switch(letter)
{
    case 'A':
    case 'a': cout<<"you typed A\n"; break;
    case 'B':
    case 'b': cout<<"you typed B\n"; break;
    default:  cout<<"Invalid input\n";
}
```

## Tracing (6)

What is the output of the following program?

```
#include <iostream>
using namespace std;
int main()
{
    int sum=0, index;
    for(index = 0; index < 30; index++);
        sum += index ;
        cout <<sum ;
}
```

There is ; after the loop so the loop will be executed 30 times

index now = 30 will be added to sum, so the output = 30

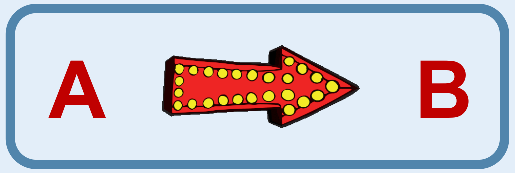
## Tracing (6) – cont’d

After how many times will the following loop terminate?

```
#include <iostream>
using namespace std;
int main()
{
    int sum=0;
    int n=10;
    while (n<=100)
        sum+= n*n ;
}
```

**Never… n is not modified anywhere**

## Type Conversion



## Tracing (7)

What will be the output?

```
#include <iostream>
using namespace std;
int main()
{
		int count = 7 ;
		float avgweight = 155.5;
		double totalweight = count * avgweight ;
		cout << “total weight =” << totalweight << endl ;
}
```

| Data Type | Order |
|---|---|
| long double | Highest |
| double | |
| float | |
| long | |
| int | |
| short | |
| char | Lowest |

# Tracing (8)

What will be the output?

```
#include <iostream>
using namespace std;
int main()
{
    int nValue1 = 10;
    int nValue2 = 4;
    float fValue = nValue1 / nValue2;
    cout <<fValue;
}
```

fValue = 2

## Type Conversion

How can we correct this mistake?

```
#include <iostream>
using namespace std;
int main()
{
    int nValue1 = 10;
    int nValue2 = 4;
    float fValue = (float)nValue1
    cout <<fValue;
}
```

fValue = 2.5

## const
## and
## #define

## Develop! (1)

Write a program to output the ***area*** of a ***circle*** where ***PI*** is defined once as:

1. a **const** qualifier

2. a **#define** directive

## Develop! Solution… (1)

1. ***const*** qualifier:

```
#include <iostream>
using namespace std;
int main()
{
    const float PI = 3.14159;
    float radius;
    cout << "Enter the radius : ";
    cin >> radius;
    cout << "Area = " << PI * radius * radius <<endl;
}
```

# Develop! Solution… (1)

2. ***#define*** directive:

```
#include <iostream>
using namespace std;
#define PI 3.14159 /* note: no datatype, no semicolon, no
equal*/

int main()
{
    float radius;
    cout << "Enter the radius : ";
    cin >> radius;
    cout << "Area = " << PI * radius * radius <<endl;
}
```

# Develop! (2)

Write a program to determine whether a given number is a ***prime***.

*(A prime number is only divisible by 1 and itself)*

**Sample Execution:**

```
Enter a number: 4
Not prime!

Enter a number: 7
Prime!
```

**Use:**

1- `break` in the loop body

2- `boolean` variable as a flag

```
int Number;
bool prime = true ;
cout<< "enter number: ";
cin>> Number;
if(Number==1)
    cout<< " 1 is not prime " << endl;
else
if(if(Number==2)
    cout<< " 2 is prime " << endl;
else
{
    for (int i=2;i<Number;i++)
    {
        if (Number%i==0)
        {
            prime=false;
            break;
        }
    }
    if (prime == true)
            cout<<Number<<" is a prime\n";
    else
            cout<<Number<<" is not a prime\n";
}
```

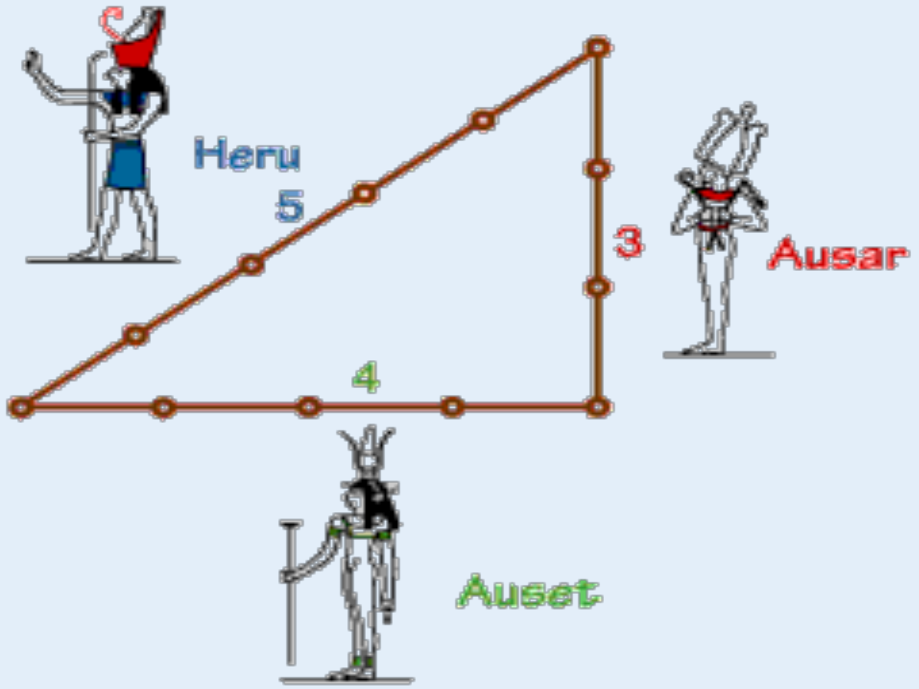
## Practice

Problem 11854 in UVa

Open Problem

## Develop! (3)

- A long time ago, the Egyptians figured out that a triangle with sides of length 3, 4, and 5 had a right angle as its largest angle.
- You must determine if other triangles have a similar property.



## Develop (3) – cont’d

**Input:**

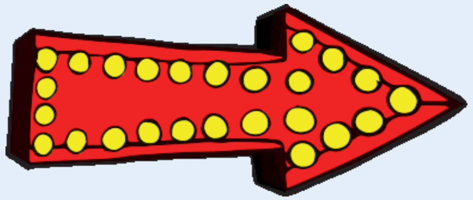
- Input reads several test cases, followed by a line containing 0 0 0.
- Each test case has three positive integers, less than 30,000, denoting the lengths of the sides of a triangle.

**Output:**

- For each test case, a line containing "right" if the triangle is a right triangle, and a line containing "wrong" if the triangle is not a right triangle.

## Develop (3) – cont’d

| **Sample Input:** | **Output for Sample Input:** |
|---|---|
| 6 8 10 | right |
| 25 52 60 | wrong |
| 5 12 13 | right |
| 0 0 0 | |



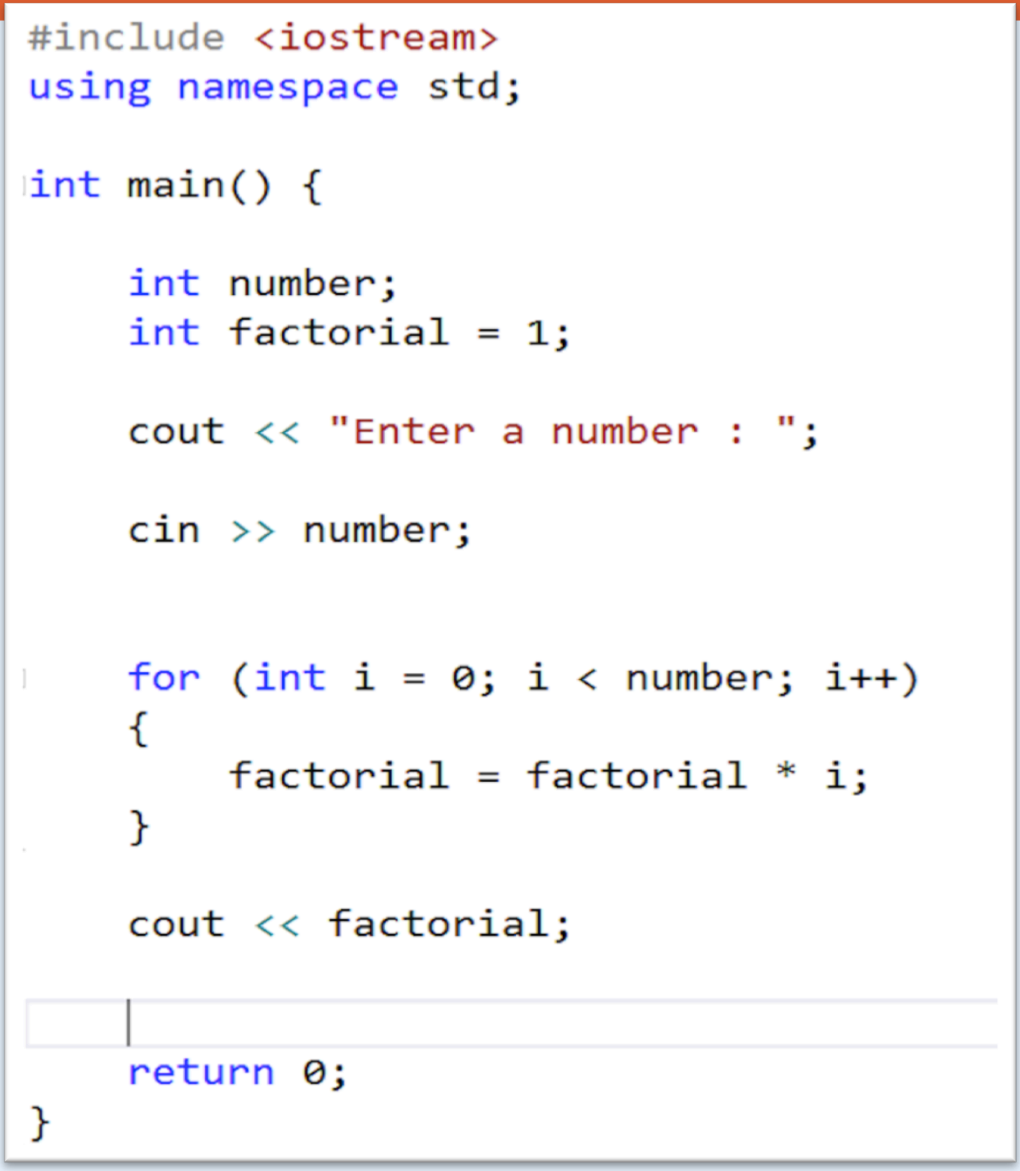
```cpp
int s1, s2, s3;
while(true)
{
    cin>>s1>>s2>>s3;
    if(s1 == 0 && s2 == 0 && s3 == 0)
        break;
    if(s1 < 0 || s2 < 0 || s3 < 0)
        cout<<"wrong\n";
    else if(s1*s1 + s2*s2 == s3*s3)
        cout<<"right\n";
    else if(s2*s2 + s3*s3 == s1*s1)
        cout<<"right\n";
    else if(s3*s3 + s1*s1 == s2*s2)
        cout<<"right\n";
    else
        cout<<"wrong\n";
}
```

## Debugging Example

Finding the Factorial of a Number

## Debugging Example

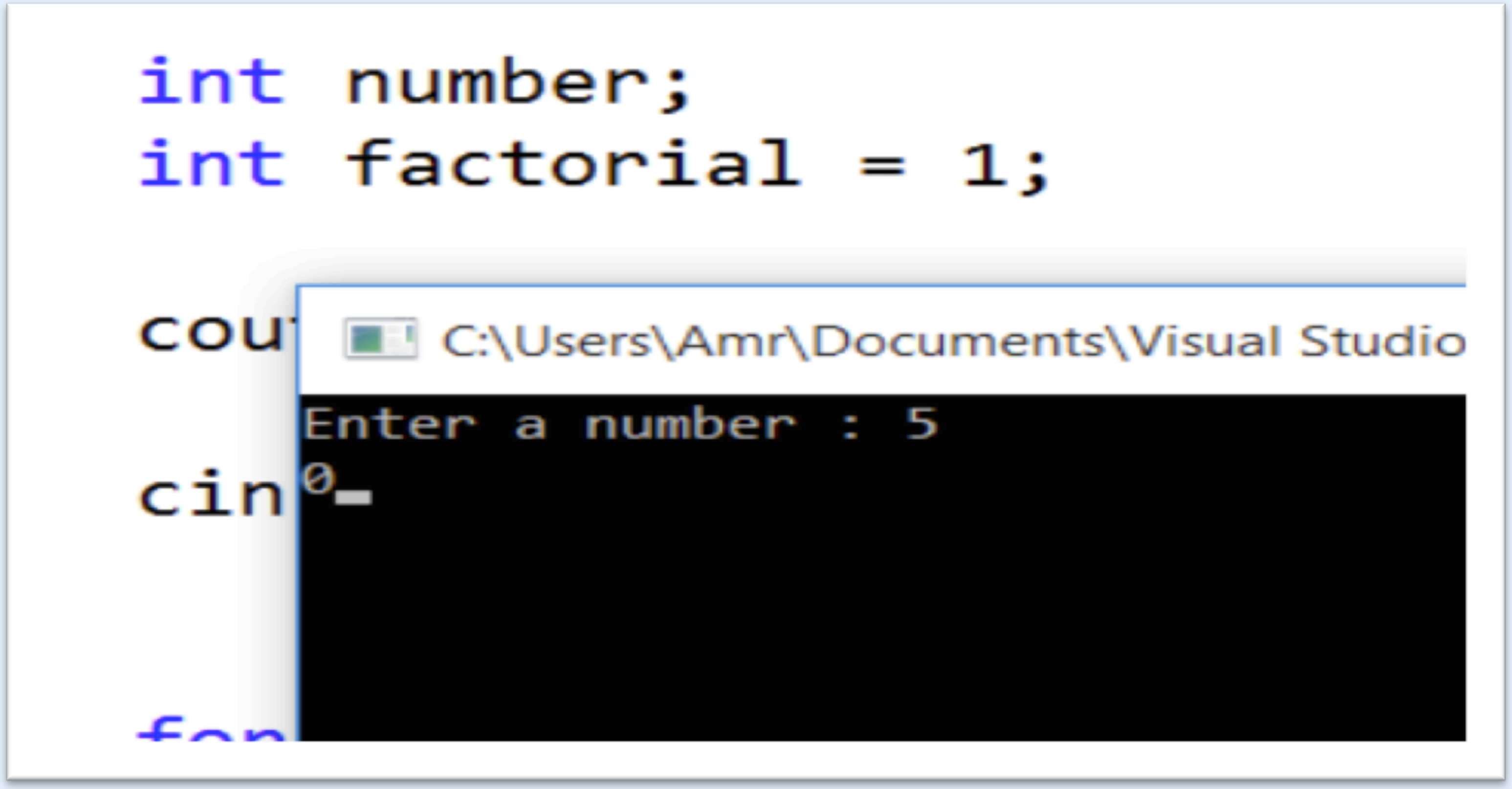
Write a program to calculate the factorial of a number provided by the user.



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

int main() {

    int number;
    int factorial = 1;

    cout << "Enter a number : ";

    cin >> number;


    for (int i = 0; i < number; i++)
    {
        factorial = factorial * i;
    }

    cout << factorial;


    return 0;
}
```

# Debugging Example – cont.

Wrong answer!



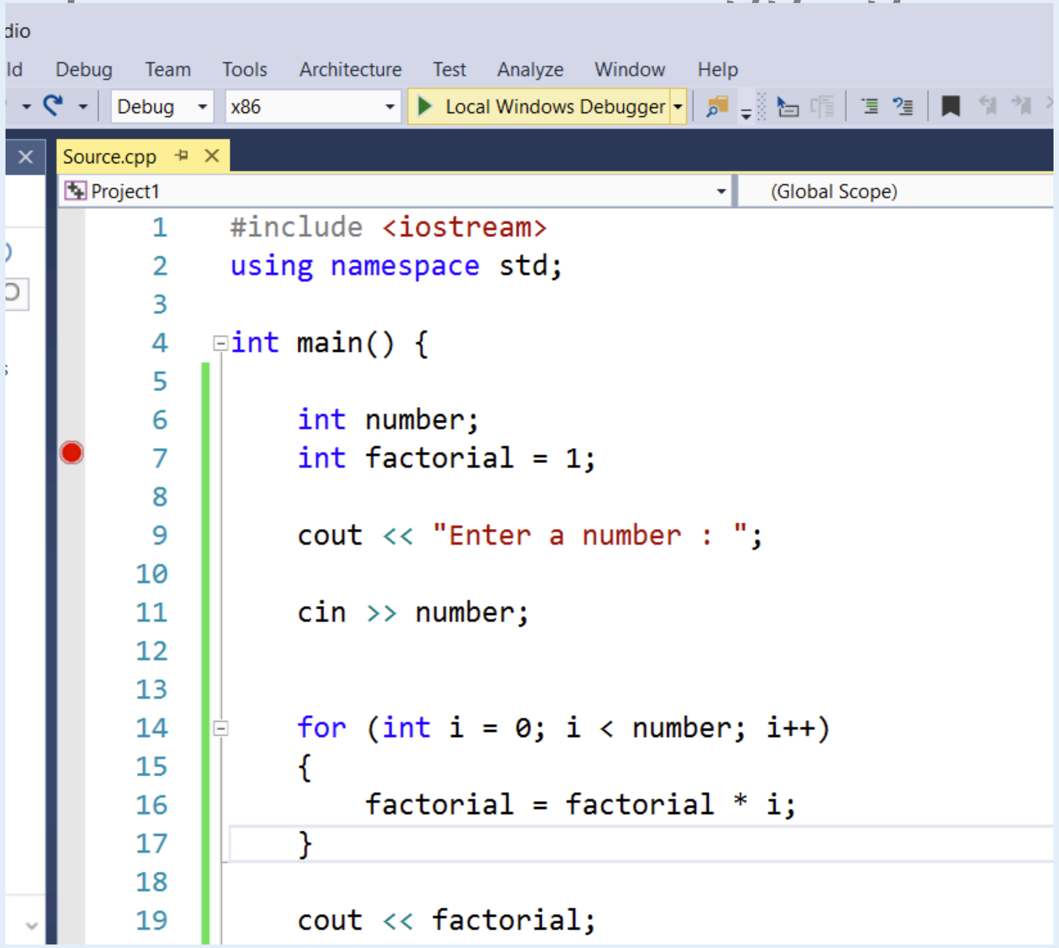
```
int number;
int factorial = 1;

cou

cin

for
```

C:\Users\Amr\Documents\Visual Studio

```
Enter a number : 5
0_
```

# Debugging Example – cont.

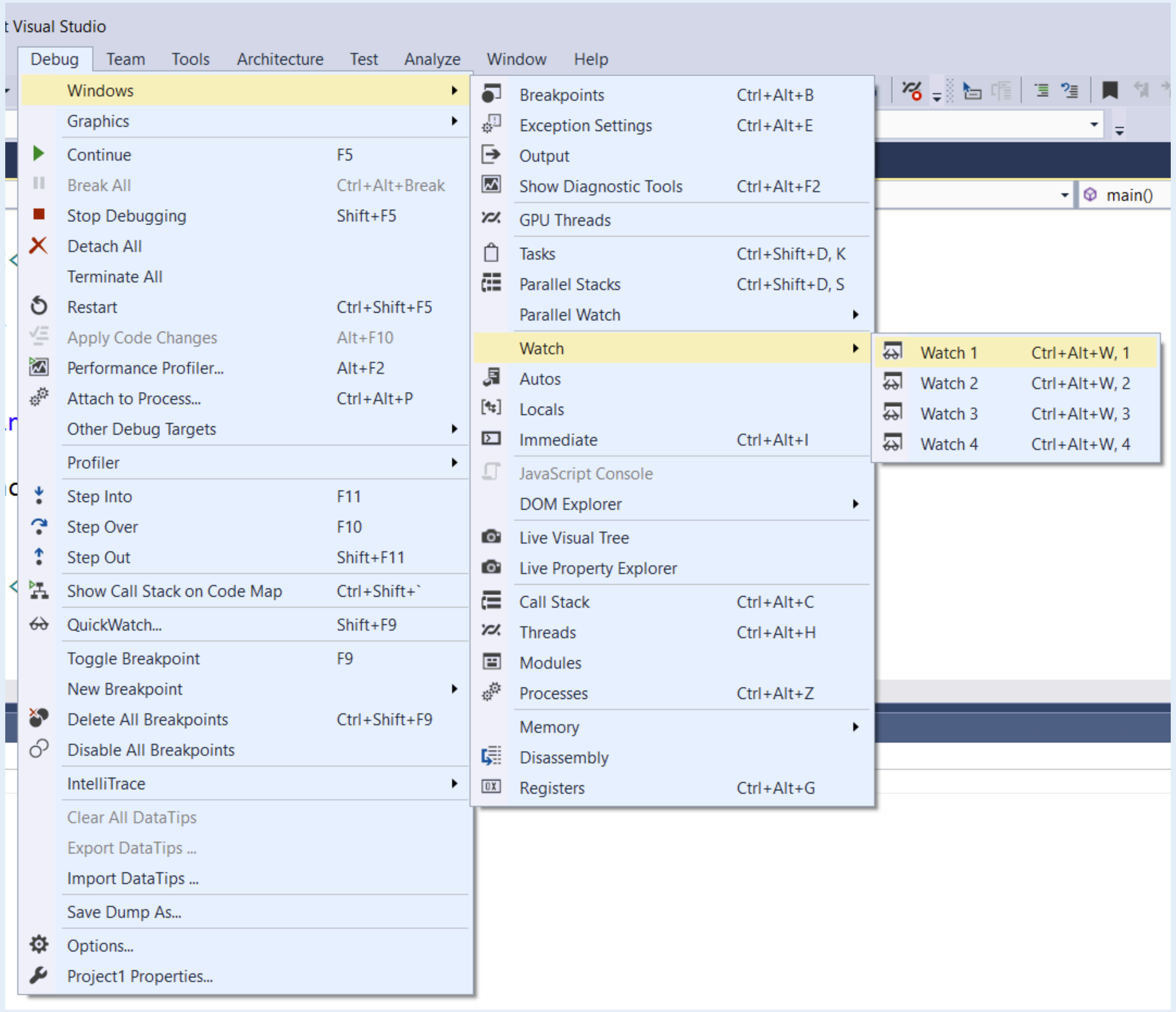
Set a Breakpoint and start debugging



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

int main() {

    int number;
    int factorial = 1;

    cout << "Enter a number : ";

    cin >> number;


    for (int i = 0; i < number; i++)
    {
        factorial = factorial * i;
    }

    cout << factorial;
```

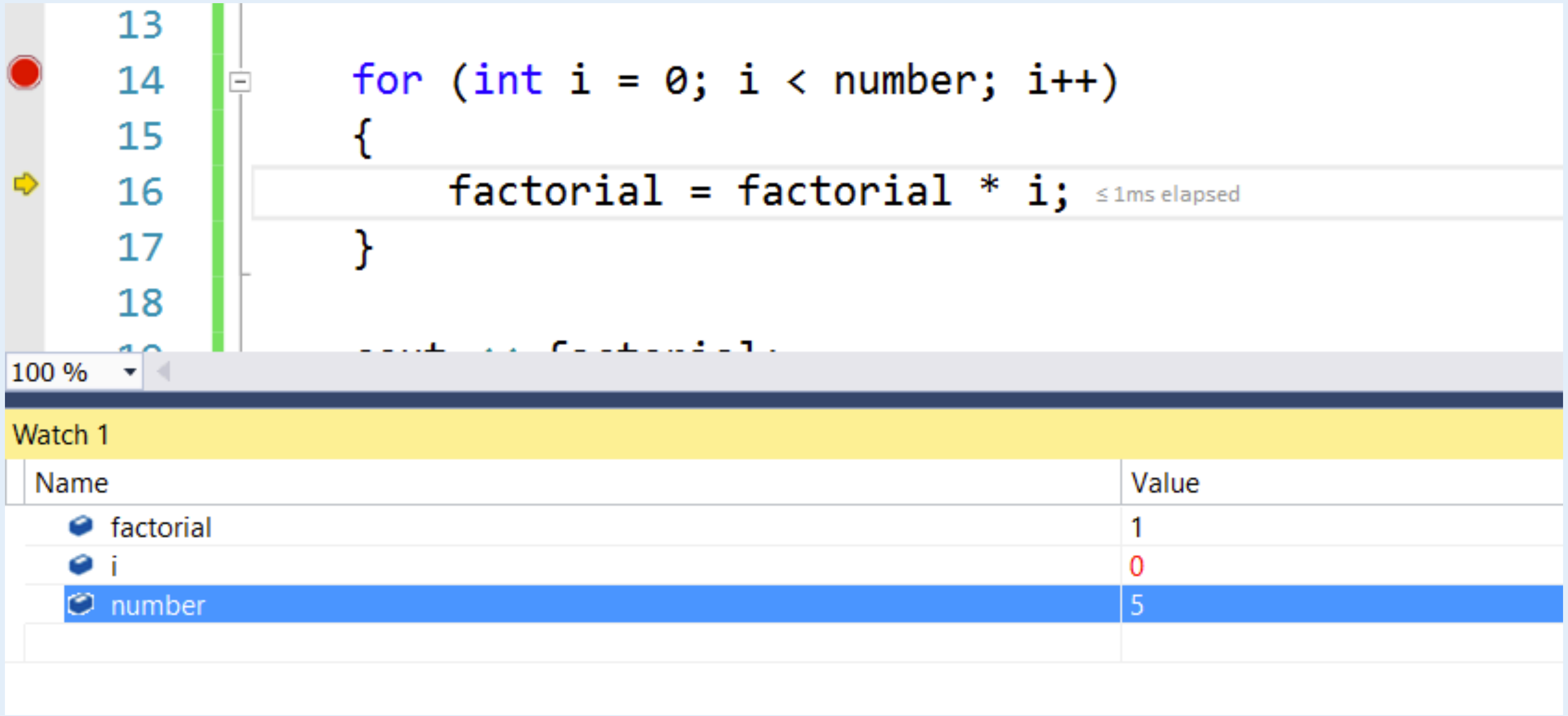
# Debugging Example – cont.

Create a Watch and add all the variables you want to trace.



# Debugging Example – cont.

- Step Over the code and observe the values in the Watch Window



```
    for (int i = 0; i < number; i++)
    {
        factorial = factorial * i;  ≤ 1ms elapsed
    }
```

100 %

Watch 1

| Name | Value |
|---|---|
| factorial | 1 |
| i | 0 |
| number | 5 |

## Debugging Example – cont.

- Find the problem

```
for (int i = 0; i < number; i++)
{
        factorial = factorial * i;
}
```

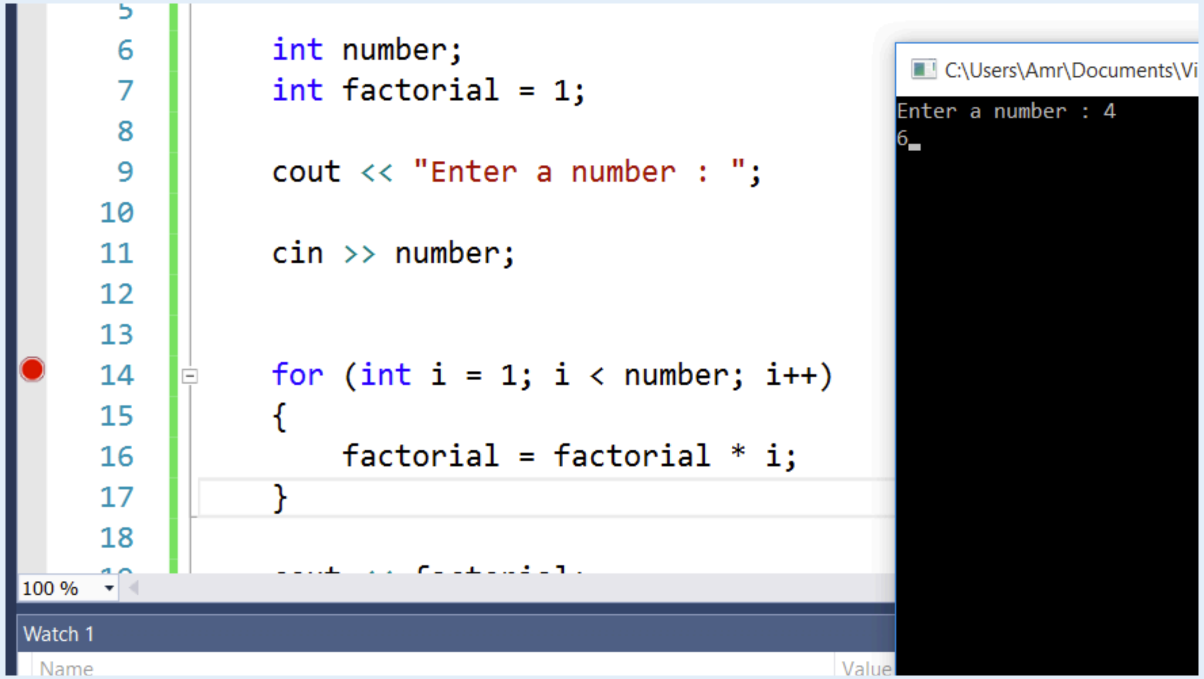
`int i = 0`

`i` can not start with `0;`

(Multiplying by zero makes a wrong solution)

## Debugging Example – cont.

Fix and Restart Debugging



```
int number;
int factorial = 1;

cout << "Enter a number : ";

cin >> number;


for (int i = 1; i < number; i++)
{
    factorial = factorial * i;
}
```

```
Enter a number : 4
6
```

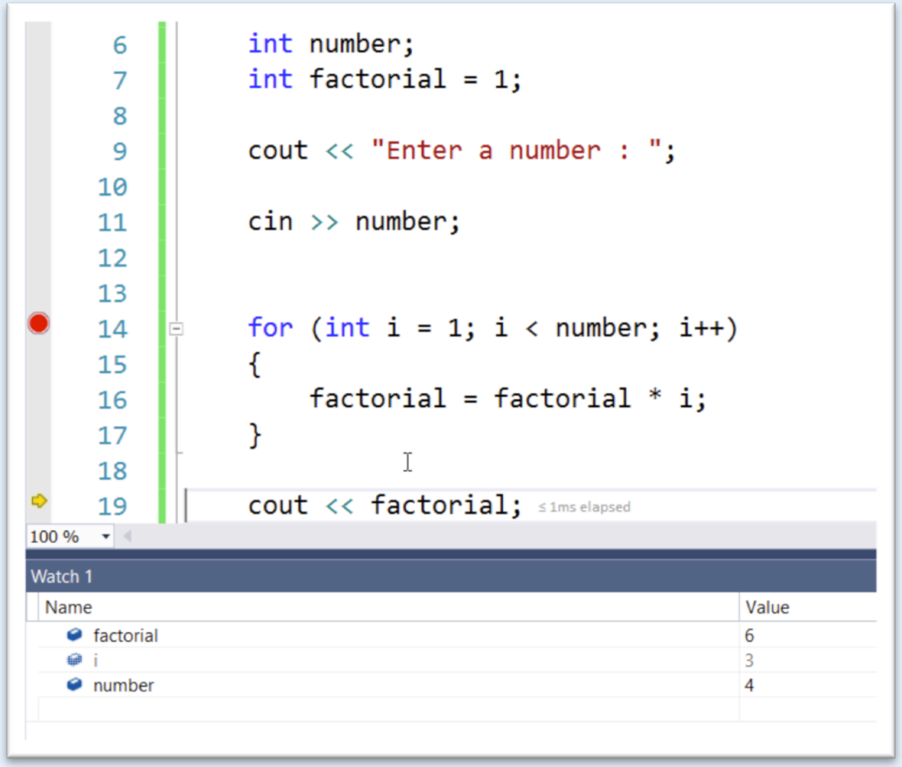
Wrong Answer    Factorial 4 = 24 not 6

# Debugging Example – cont.

Trace the variable values in the Watch window

What is the problem?

The loop finishes before multiplying the last number



```
int number;
int factorial = 1;

cout << "Enter a number : ";

cin >> number;


for (int i = 1; i < number; i++)
{
    factorial = factorial * i;
}

cout << factorial;  ≤ 1ms elapsed
```

100 %

Watch 1

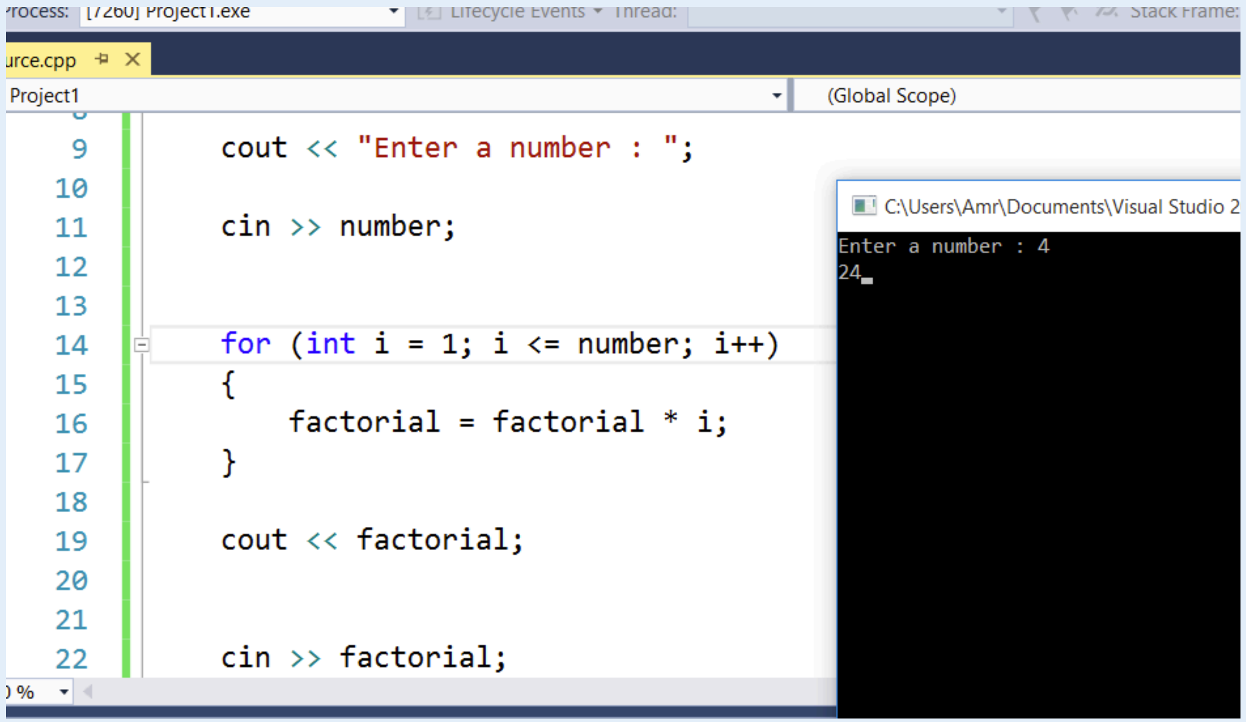
| Name | Value |
|---|---|
| factorial | 6 |
| i | 3 |
| number | 4 |

```
for (int i = 1; i < number; i++)
```

- The loop condition must be less than or equal **(<=)** not less than **(<)**

# Debugging Example – cont.

Fix and Retry


Process: [7260] Project1.exe
Lifecycle Events
Thread:
Stack Frame:
urce.cpp
Project1
(Global Scope)
C:\Users\Amr\Documents\Visual Studio 2

```
    cout << "Enter a number : ";

    cin >> number;


    for (int i = 1; i <= number; i++)
    {
        factorial = factorial * i;
    }

    cout << factorial;


    cin >> factorial;
```

```
Enter a number : 4
24
```

- Correct Answer

# Thank you!