

## Module 8 Functions

### 2. Function Definition & Execution Model

#### 2.1 CS Bridge: Functions



NYU TANDON  
ONLINE

## CS Bridge: Functions

Module 8  
Itay Tal

#### 2.2 k-combinations

### K-combinations Problem

NYU TANDON  
ONLINE

**Definition:** Let  $n$  and  $k$  be two nonnegative integers, such that  $k \leq n$ . We define  **$k$ -combinations** as the number of unordered selections of  $k$  distinct elements.  $k$ -combinations is denoted by  $\binom{n}{k}$  and is also called  **$n$  choose  $k$** .

$n = 5$        $k = 3$

$$\binom{5}{3} = \frac{5!}{3! \cdot 2!} = 10$$

3-combinations



**Theorem:** Let  $n$  and  $k$  be two nonnegative integers, such that  $k \leq n$ . The number of  $k$ -combinations of a set with  $n$  elements equals:  $\binom{n}{k} = \frac{n!}{k! \cdot (n-k)!}$

## 2.3 k-combination Problem

### K- Combinations Problem

NYU TANDON  
ONLINE

Problem

Write a program that reads from the user two positive integers n, k ( $n \geq k$ ), and prints the value of  $n$  choose  $k$ .

Example

Please enter n and k ( $k \leq n$ ):  
5 3  
5 choose 3 is 10



## 2.4 k-Combination Solution

```
int main(){
    int n, k,
    cout<<"Please enter n and k (n>=k):"<<endl;
    cin>>n>>k;

    nFact = 1;
    nFact<=factorial(p);
    nFact *= i;

    kFact = 1;
    kFact<=factorial(k);
    kFact *= i;

    n_kFact = 1;
    n_kFact<=factorial(p-k);
    n_kFact *= i;

    k_comb = nFact / (kFact*n_kFact);
    cout<<n<<" choose "<<k<<" is "<<k_comb<<endl;
    return 0;
}
```

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

NYU TANDON  
ONLINE



## 2.5 Flow of a Program That Calls Functions

### Flow of a Program That Calls Functions

NYU TANDON  
ONLINE

```
int factorial(int num){
    int factRes, i;

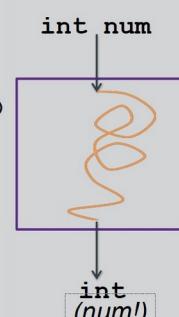
    factRes = 1;
    for (i=1; i<=num; i++)
        factRes *= i;

    return factRes;
}
```

int num

factorial

int (num!)



## 2.6 Calling a Function Code

```

int main() {
    int n, k, k_comb;
    int nFact, kFact, n_kFact;
    cout<<"Please enter n and k "<<endl;
    cin>>n>>k;
    nFact = factorial(n);
    kFact = factorial(k);
    n_kFact = factorial(n-k);
    k_comb = nFact / (kFact*n_kFact);
    cout<<n<<" choose "<<k<<" is ";
    cout<<k_comb<<endl;
    return 0;
}
int factorial(int num) {
    int factRes, i;
    factRes = 1;
    for (i=1; i<=num; i++)
        factRes *= i;
    return factRes;
}

```

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

## 2.7 Data Types, Expressions, Control Flow

Data	Expressions	Control Flow
<ul style="list-style-type: none"> <li>int</li> <li>float</li> <li>double</li> <li>char</li> <li>string</li> <li>bool</li> </ul>	<ul style="list-style-type: none"> <li>I/O expressions</li> <li>Arithmetic expressions</li> <li>Boolean expressions</li> </ul>	<ul style="list-style-type: none"> <li>Sequential</li> <li>Branching           <ul style="list-style-type: none"> <li>if</li> <li>if-else</li> <li>if-else if-else</li> <li>switch</li> </ul> </li> <li>Iterative           <ul style="list-style-type: none"> <li>while</li> <li>for</li> </ul> </li> <li>Function call</li> </ul>

## 2.8 Runtime Stack Execution Model

```

1. int main() {
2.     int n, k, k_comb;
3.     int nFact, kFact, n_kFact;
4.     cout<<"Please enter n and k "<<endl;
5.     cin>>n>>k;
6.     nFact = factorial(n);
7.     kFact = factorial(k);
8.     n_kFact = factorial(n-k);
9.     k_comb = nFact / (kFact*n_kFact);
10.    cout<<n<<" choose "<<k<<" is ";
11.    cout<<k_comb<<endl;
12.    return 0;
13. }
14.
15. int factorial(int num){
16.     int factRes, i;
17.     factRes = 1;
18.     for (i=1; i<=num; i++)
19.         factRes *= i;
20.     return factRes;
21. }

```

The stack diagram shows the state of variables in the main() function frame and the factorial() function frame. In the main() frame, n=5, k=3, k\_comb is undefined, nFact=120, kFact is undefined, and n\_kFact is undefined. In the factorial() frame, num=5, factRes=120, i=6, and return adrs=6.

## 2.9 Runtime Stack Execution Model

```

1. int main(){
2.     int n, k, k_comb;
3.     int nFact, kFact, n_kFact;
4.     cout<<"Please enter n and k "<<endl;
5.     cin>>n>>k;
6.     nFact = factorial(n);
7.     kFact = factorial(k);
8.     n_kFact = factorial(n-k);
9.     k_comb = nFact / (kFact*n_kFact);
10.    cout<<n<<" choose "<<k<<" is ";
11.    cout<<k_comb<<endl;
12.    return 0;
13. }
14.
15. int factorial(int num){
16.     int factRes, i;
17.     factRes = 1;
18.     for (i=1; i<=num; i++)
19.         factRes *= i;
20.     return factRes;
21. }

```

## 2.10 Program Implementation

## 2.11 Scope of Variables Example

```

1. int main(){
2.     int n = 3;
3.     cout<<"Before func:"<<n<<<endl;
4.     func(n);
5.     cout<<"After func:"<<n<<<endl;
6.     return 0;
7. }
8.
9. void func(int n){
10.    n = 4;
11.    cout<<"Inside func:"<<n<<<endl;
12. }

```

Before func: 3  
Inside func: 4  
After func: 3

# 1. Breaking a Program into Functions

## 1.1 Solving Quadratic Equation

Solving Quadratic Equations

NYU TANDON  
ONLINE

Problem  $(ax^2 + bx + c = 0, a \neq 0) \rightarrow x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$

Write a program that reads 3 real numbers, representing coefficients of a quadratic equation, and prints the solutions of the equation, if there are any, or an appropriate message.

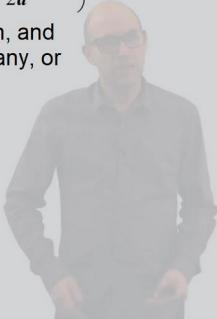
- Two Real Solutions  
E.g.  $x^2 - 5x + 6 = 0$        $x_1 = 2, x_2 = 3$
- One Real Solution  
E.g.  $x^2 + 2x + 1 = 0$        $x = -1$
- No Solution  
The equation:  $1x^2 + -5x + 6 = 0$

Please enter coefficients of quadratic equation:

1 -5 6

The equation:  $1x^2 + -5x + 6 = 0$

Solutions: 2 3



## 1.2 Implementation 1

Implementation

NYU TANDON  
ONLINE

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

// Constants to represent type of solutions to an equation
const int NO SOLUTION = 0;
const int ONE_REAL SOLUTION = 1;
const int TWO_REAL SOLUTIONS = 2;
const int ALL_REALS = 3;
const int NO_REAL SOLUTION = 4;

// quadratic: Solves the quadratic equation: ax^2+bx+c = 0
// Input:
// Output:
// Assumptions:
quadratic(double a, double b, double c,
          double& outX1, double& outX2);
```



## 1.3 Implementation 2

```
// This program solves a quadratic equation.  
// Input from user: 3 real numbers, representing coefficients of a quadratic equation  
// Output to user: The solutions of the equation, if there are any, or an appropriate message  
int main() {  
    double a, b, c;  
  
    cout << "Please enter coefficients of quadratic equation:\n";  
    cin >> a >> b >> c;  
    cout << "The equation: " << a << "x^2 + " << b << "x + " << c << " = 0" << endl;  
    switch (quadratic(a, b, c, x1, x2)) {  
        case TWO_REAL_SOLUTIONS:  
            cout << "Solutions: " << x1 << " " << x2 << endl; break;  
        case ONE_REAL SOLUTION:  
            cout << "One solution: " << x1 << endl; break;  
        case NO_REAL_SOLUTION:  
            cout << "No real solution" << endl; break;  
        case ALL_REALS:  
            cout << "All real numbers are solutions" << endl; break;  
        default:  
            cout << "Error" << endl; break;  
    }  
    return 0;  
}
```



## 1.4 Implementation 3

```
// This program solves a quadratic equation.  
// Input from user: 3 real numbers, representing coefficients of a quadratic equation  
// Output to user: The solutions of the equation, if there are any, or an appropriate message  
#include <iostream>  
#include <cmath>  
using namespace std;  
  
// Constant: Please enter coefficients of quadratic equation:  
const int NO_SOLUTION = 0;  
const int ONE_REAL SOLUTION = 1;  
const int TWO_REAL SOLUTIONS = 2;  
const int ALL_REALS = 3;  
const int TWO_REAL SOLUTION:  
    cout << "Solutions: " << x1 << " " << x2 << endl; break;  
case ONE_REAL SOLUTION:  
    cout << "One solution: " << x1 << endl; break;  
// Input case NO_REAL_SOLUTION:  
// Output: 1. cout << "No real solution" << endl; break;  
// case NO_REAL_SOLUTION2 - solutions to equation [output parameters]  
// Assumption: cout << "No solutions" << endl; be breaked in outX1  
// case ALL_REALS: equation has no real solutions the values in outX1 and outX2 are not defined  
int quadratic(double a, double b, double c, double& outX1, double& outX2) {  
    const double delta = b * b - 4 * a * c;  
    if (delta <= 0.0) {  
        if (delta == 0.0) {  
            outX1 = -b / (2 * a);  
            outX2 = -b / (2 * a);  
            return ONE_REAL SOLUTION;  
        }  
        else  
            return NO_REAL_SOLUTION;  
    }  
    else  
        return TWO_REAL SOLUTION;  
}
```



## 1.5 Implementation 4

```
int quadratic(double a, double b, double c,  
             double& outX1, double& outX2) {  
    double delta;  
    if (a != 0.0){  
        delta = b*b - 4*a*c;  
        if (delta > 0){  
            x1 = (-b + sqrt(delta))/(2*a);  
            x2 = (-b - sqrt(delta))/(2*a);  
            outX1 = x1;  
            outX2 = x2;  
            return TWO_REAL SOLUTION;  
        }  
        else if (delta == 0){  
            x1 = -b/(2*a);  
            outX1 = x1;  
            return ONE_REAL SOLUTION;  
        }  
        else  
            return NO_REAL_SOLUTION;  
    }  
}
```

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$



## 1.6 Implementation 5

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

// Constants to represent type of solutions to an equation
const int NO SOLUTION = 0;
const int ONE_REAL SOLUTION = 1;
const int TWO_REAL SOLUTIONS = 2;
const int ALL_REALS = 3;
const int NO_REAL SOLUTION = 4;

// quadratic: Solves the quadratic equation: ax^2+bx+c = 0
// Input: a, b, c - coefficients of equation
// Output: 1. Type of solution (return value)
//         2. outX1, outX2 - solutions to equation (output parameters)
// Assumptions: 1. If equation has one solution it will be returned in outX1
//             2. If equation has no real solutions the values in outX1 and outX2 are not defined
int quadratic(double a, double b, double c,
              double& outX1, double& outX2);

// linear: Solves a linear equation: ax+b = 0
// Input: a, b - coefficients of equation
// Output: 1. Type of solution (return value)
//         2. outX - solution to equation (output parameter)
// Assumptions: If equation has no solutions the value returned at outX is not defined
linear();
```



## 1.7 Implementation 6

```
int quadratic(double a, double b, double c,
              double& outX1, double& outX2) {
    double delta, x1, x2;
    if (a != 0.0) {
        delta = b*b - 4*a*c;
        if (delta > 0) {
            x1 = (-b + sqrt(delta))/(2*a);
            x2 = (-b - sqrt(delta))/(2*a);
            outX1 = x1;
            outX2 = x2;
            return TWO_REAL_SOLUTION;
        }
        else if (delta == 0) {
            x1 = -b/(2*a);
            outX1 = x1;
            return ONE_REAL_SOLUTION;
        }
        else
            return NO_REAL_SOLUTION;
    }
    else
        linear();
}
```



## 1.8 Implementation 7

$ax + b = 0$

```
int linear(double a, double b, double& outX) {
    double x;
    if (a != 0) {
        x = -b/a;
        outX = x;
        return ONE_REAL_SOLUTION;
    }
    else if ((a == 0) && (b == 0)) {
        x = 0;
        outX = x;
        return ALL_REALS;
    }
    else // in this case a==0 && b!=0
        return NO SOLUTION;
}
```



## 1.9 Knowledge Check

(True/False, 10 points, unlimited attempts permitted)

Knowledge Check

NYU TANDON  
ONLINE

Suppose that in a function, the parameters are passed in by reference. After this function changes the value of each parameter, what happens to the values of the parameters outside the function?

Nothing, the values of each parameter remain unchanged  
 The values change according to what the function has changed them to

Correct	Choice
	Nothing, the values of each parameter remain unchanged
X	The values change according to what the function has changed them to

**Feedback when correct:**

That's right! You selected the correct response.

## Correct (Slide Layer)

Knowledge Check

NYU TANDON  
ONLINE

Suppose that in a function, the parameters are passed in by reference. After this function changes the value of each parameter, what happens to the values of the parameters outside the function?

Correct  
That's right! You selected the correct response.

Nothing, the values of each parameter remain unchanged  
 The values change according to what the function has changed them to

## Try Again (Slide Layer)

Knowledge Check  NYU TANDON  
ONLINE

Suppose that in a function, the parameters are passed in by reference. After this function changes the value of each parameter outside the function, what happens to the values of the parameters?

Incorrect

That is incorrect. Please try again.

Try Again

Nothing, the values of each parameter remain unchanged  
 The values change according to what the function has changed them to

## 1.10 Knowledge Check

(Multiple Choice, 10 points, 4 attempts permitted)

Knowledge Check  NYU TANDON  
ONLINE

In the following code, what are the values of a and b after the function is finished?

```
int main() {
int a = 5, b = 6;
swap(a,b);
return 0;
}
void swap(int a, int& b){
    int temp;
    temp = a;
    a = b;
    b = temp}
```

a=6 , b=5  
 a=5, b=5  
 a=6, b=6  
 a=5, b=6

Correct	Choice	Feedback
	a=6 , b=5	If both a and b were passed in by reference, then this would be correct!
X	a=5, b=5	Correct! Note that b is passed in by reference, so its value inside main changes, unlike a, which is passed in by value.

a=6, b=6

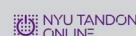
Recall that a is passed in by value, so  
the value of it inside the main function  
doesn't change!

a=5, b=6

Recall that b is passed in by reference,  
so the value of it inside the main  
function changes!

## a=6 , b=5 (Slide Layer)

### Knowledge Check



In the following code, what are the values of a and b after  
the function is finished?

```
int main() {  
    int a = 5, b;  
    swap(a,b);  
    return 0;  
}  
void swap(int  
         int temp;  
         temp = a;  
         a = b;  
         b = temp}
```

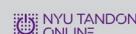
Incorrect

If both a and b were passed in by reference, then this  
would be correct!

Continue

## a=5, b=5 (Slide Layer)

### Knowledge Check



In the following code, what are the values of a and b after  
the function is finished?

```
int main() {  
    int a = 5, b;  
    swap(a,b);  
    return 0;  
}  
void swap(int  
         int temp;  
         temp = a;  
         a = b;  
         b = temp}
```

Correct

Correct! Note that b is passed in by reference, so its value  
inside main changes, unlike a, which is passed in by value.

Continue

## a=6, b=6 (Slide Layer)

Knowledge Check

NYU TANDON  
ONLINE

In the following code, what are the values of a and b after the function is finished?

```
int main() {
    int a = 5, b;
    swap(a,b);
    return 0;
}
void swap(int
    int temp
    temp = a
    a = b;
    b = temp)
```

Incorrect

Recall that a is passed in by value, so the value of it inside the main function doesn't change!

Continue

## a=5, b=6 (Slide Layer)

Knowledge Check

NYU TANDON  
ONLINE

In the following code, what are the values of a and b after the function is finished?

```
int main() {
    int a = 5, b;
    swap(a,b);
    return 0;
}
void swap(int
    int temp
    temp = a
    a = b;
    b = temp)
```

Incorrect

Recall that b is passed in by reference, so the value of it inside the main function changes!

Continue

## 1.11 Knowledge Check

(Multiple Choice, 10 points, 6 attempts permitted)

## Knowledge Check



What does the “&” symbol represent?

- It's a parameter
- And
- Passed by Value
- Passed by Reference
- Concatenation

Correct	Choice	Feedback
	It's a parameter	Almost! The parameter is the variable that's passed in, but the & symbol refers to how it's passed in.
	And	Not quite! The And symbol is represented by && in C++
	Passed by Value	Not quite! The & symbol allows values to change outside the function while a parameter passed by value creates a “copy” that only changed within the function.
X	Passed by Reference	Yes! Remember, this means that the value will change even outside of the function.
	Concatenation	Remember, concatenation is represented by + in C++

## It's a parameter (Slide Layer)

Knowledge Check

NYU TANDON  
ONLINE

What does the “&” symbol represent?

It's a parameter

And

Passed by

Passed by

Concatenation

Incorrect

Almost! The parameter is the variable that's passed in, but the & symbol refers to how it's passed in.

[Continue](#)

## And (Slide Layer)

Knowledge Check

NYU TANDON  
ONLINE

What does the “&” symbol represent?

It's a parameter

And

Passed by

Passed by

Concatenation

Incorrect

Not quite! The And symbol is represented by && in C++

[Continue](#)

## Passed by Value (Slide Layer)

Knowledge Check

NYU TANDON  
ONLINE

What does the “&” symbol represent?

It's a parameter

And

Passed by

Passed by

Concatenation

Incorrect

Not quite! The & symbol allows values to change outside the function while a parameter passed by value creates a “copy” that only changed within the function.

[Continue](#)

## Passed by Reference (Slide Layer)

Knowledge Check

NYU TANDON  
ONLINE

What does the “&” symbol represent?

It's a parameter  
 And  
 Passed by  
 Passed by reference  
 Concatenation

Correct  
Yes! Remember, this means that the value will change even outside of the function.

Continue

## Concatenation (Slide Layer)

Knowledge Check

NYU TANDON  
ONLINE

What does the “&” symbol represent?

It's a parameter  
 And  
 Passed by  
 Passed by reference  
 Concatenation

Incorrect  
Remember, concatenation is represented by + in C++

Continue

## 1.12 Knowledge Check

(Multiple Choice, 10 points, 4 attempts permitted)

## Knowledge Check



What statement is required at the end of every function that wished to return a value?

- To be called
- A return statement
- The create variables
- Parameters

Correct	Choice	Feedback
	To be called	This is not a requirement. It can also potentially cause an infinite loop if a certain statement is missing!
X	A return statement	Correct! Remember that the only time a function doesn't need to return something is if it's declared as void.
	The create variables	Not quite! Your function can use the parameters passed in to do all the work required without ever creating new variables.
	Parameters	Not quite! Parameters are inserted before a function even begins. If you wish to save changes to a parameter, a specific keyword is needed at the end of a function.

## A return statement (Slide Layer)

Knowledge Check

NYU TANDON  
ONLINE

What statement is required at the end of every function that wished to return a value?

Correct

Correct! Remember that the only time a function doesn't need to return something is if it's declared as void.

To be called  
 A return statement  
 The create variables  
 Parameters

Continue

## To be called (Slide Layer)

Knowledge Check

NYU TANDON  
ONLINE

What statement is required at the end of every function that wished to return a value?

Incorrect

This is not a requirement. It can also potentially cause an infinite loop if a certain statement is missing!

To be called  
 A return statement  
 The create variables  
 Parameters

Continue

## The create variables (Slide Layer)

Knowledge Check

NYU TANDON  
ONLINE

What statement is required at the end of every function that wished to return a value?

Incorrect

Not quite! Your function can use the parameters passed in to do all the work required without ever creating new variables.

To be called  
 A return statement  
 The create variables  
 Parameters

Continue

## Parameters (Slide Layer)

Knowledge Check

NYU TANDON  
ONLINE

What statement is required at the end of every function that wished to return a value?

Incorrect

Not quite! Parameters are inserted before a function even begins. If you wish to save changes to a parameter, a specific keyword is needed at the end of a function.

To be called

A return statement

The create variables

Parameters

Continue

**1.13 For the following piece of code, what are the values of a and b after you exit the code?**

***int a = 1;***

***int b = 15;***

***int looperFunction(int a, int&b)***

***while ( a != b)***

***a++;***

***b--;***

(Multiple Choice, 10 points, 4 attempts permitted)

## Knowledge Check



For the following piece of code, what are the values of a and b after you exit the code?

```
int a = 1;  
int b = 15;  
int looperFunction(int a, int&b)  
{  
    while ( a != b )  
    {  
        a++;  
        b--;  
    }  
}
```

- a = 15; b = 1
- a = 1; b = 15
- a = 8; b = 8
- a = 1; b = 8

Correct	Choice	Feedback
	a = 15; b = 1	Not quite!
	a = 1; b = 15	Not quite!
	a = 8; b = 8	Almost! Notice that one parameter is pass by reference.
X	a = 1; b = 8	Correct!

## a = 1; b = 8 (Slide Layer)

## Knowledge Check



For the following piece of code, what are the values of a and b after you exit the code?

```
int a = 1;  
int b = 15;  
int looperF  
{  
    while ( a != b )  
    {  
        a++;  
        b--;  
    }  
}
```

Correct

Correct!

Continue

- a = 1; b = 8

## a = 8; b = 8 (Slide Layer)

Knowledge Check

NYU TANDON  
ONLINE

For the following piece of code, what are the values of a and b after you exit the code?

```
int a = 1;
int b = 15;
int looperF()
{
    while ( a != b )
    {
        a++;
        b--;
    }
}
```

Incorrect  
Almost! Notice that one parameter is pass by reference.

Continue

a = 1; b = 8

## a = 1; b = 15 (Slide Layer)

Knowledge Check

NYU TANDON  
ONLINE

For the following piece of code, what are the values of a and b after you exit the code?

```
int a = 1;
int b = 15;
int looperF()
{
    while ( a != b )
    {
        a++;
        b--;
    }
}
```

Incorrect  
Not quite!

Continue

a = 1; b = 8

## a = 15; b = 1 (Slide Layer)

Knowledge Check

NYU TANDON  
ONLINE

For the following piece of code, what are the values of a and b after you exit the code?

```
int a = 1;
int b = 15;
int looperF()
{
    while ( a != b )
    {
        a++;
        b--;
    }
}
```

Incorrect  
Not quite!

Continue

a = 1; b = 8

## 1.14 Results Slide

(Results Slide, 0 points, 1 attempt permitted)

The screenshot shows a results slide with the following layout:

- Results** (top left)
- NYU TANDON ONLINE** (top right)
- Your Score:** %Results.ScorePercent%% (%Results.ScorePoints% points)
- Passing Score:** %Results.PassPercent%% (%Results.PassPoints% points)
- Result:** (empty space)
- Buttons:** **Retry Quiz** and **Review Quiz** (at the bottom)

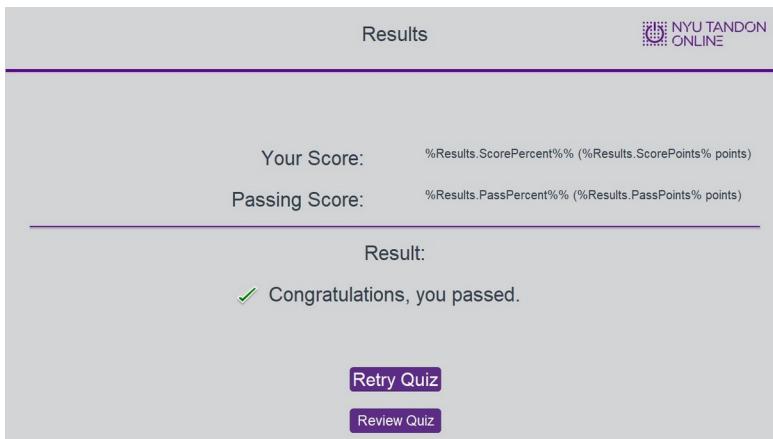
Results for
1.9 Knowledge Check
1.10 Knowledge Check
1.11 Knowledge Check
1.12 Knowledge Check
1.13 For the following piece of code, what are the values of a and b after you exit the code?  int a = 1;  int b = 15;  int looperFunction(int a, int&b)  while ( a != b )  a++;  b--;

## Result slide properties

Passing                    80%

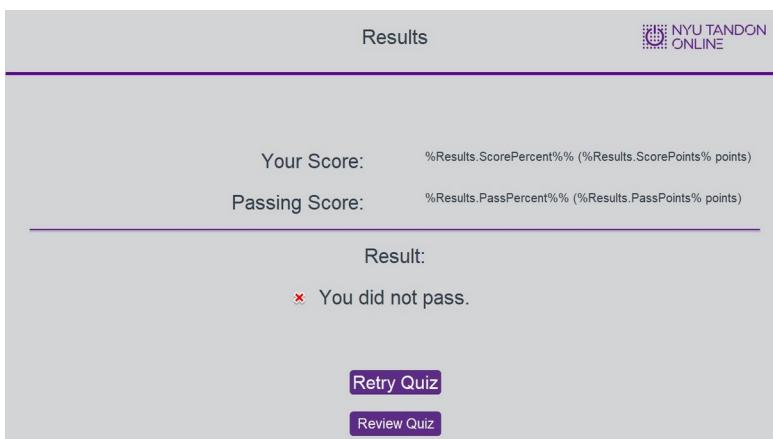
Score

## Success (Slide Layer)



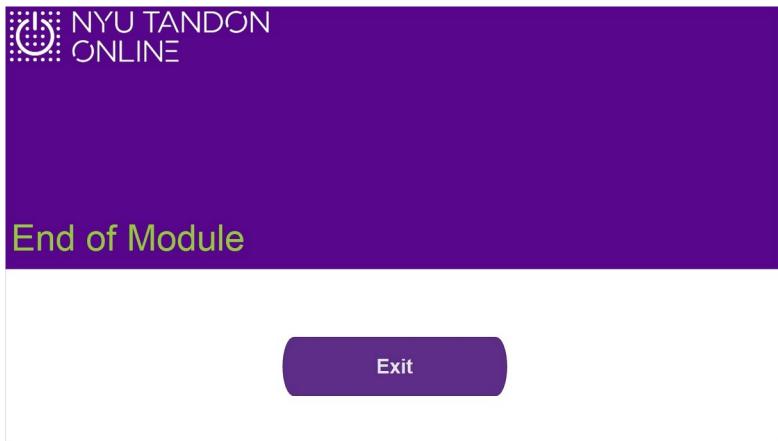
The success slide layer interface features a header with 'Results' and the NYU Tandon Online logo. Below the header, it displays 'Your Score: %Results.ScorePercent%% (%Results.ScorePoints% points)' and 'Passing Score: %Results.PassPercent%% (%Results.PassPoints% points)'. A horizontal line separates this from the result message. The result message, 'Result:', is followed by a green checkmark and the text 'Congratulations, you passed.' At the bottom are two buttons: 'Retry Quiz' and 'Review Quiz'.

## Failure (Slide Layer)



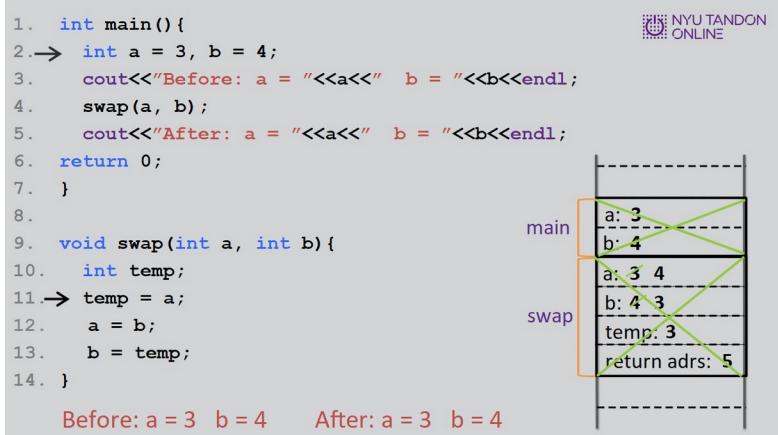
The failure slide layer interface is similar to the success one, with 'Results' and the NYU Tandon Online logo at the top. It shows 'Your Score: %Results.ScorePercent%% (%Results.ScorePoints% points)' and 'Passing Score: %Results.PassPercent%% (%Results.PassPoints% points)'. A horizontal line follows. The result message, 'Result:', is preceded by a red X and the text 'You did not pass.' At the bottom are two buttons: 'Retry Quiz' and 'Review Quiz'.

## 1.15 End of Module



## 3. Pass by Value & Pass by Reference

### 3.1 Pass By Value



## 3.2 Parameter Passing

### Parameter Passing

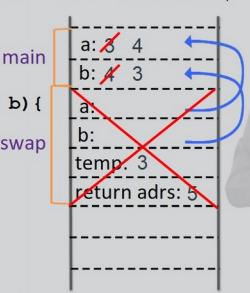
Two ways to pass parameters to a function:

- Call-By-Value
  - syntax: `void func(int x)`
  - semantics: When passing by value, the argument is evaluated and its **value** is passed
- Call-By-Reference
  - syntax: `void func(int & x)`
  - semantics: When passing by reference, a **reference** to the argument's memory location is passed



## 3.3 Example

```
1. int main(){  
2.     int a = 3, b = 4;  
3.     cout<<"Before: a = "<<a<<" b = "<<b<<endl;  
4.     swap(a, b);  
5.     cout<<"After: a = "<<a<<" b = "<<b<<endl;  
6.     return 0;  
7. }  
8.  
9. void swap(int& a, int& b){  
10.    int temp;  
11.    temp = a;  
12.    a = b;  
13.    b = temp;  
14. } Before: a = 3 b = 4  
          After: a = 4 b = 3
```



## 3.4 Analyze Digits

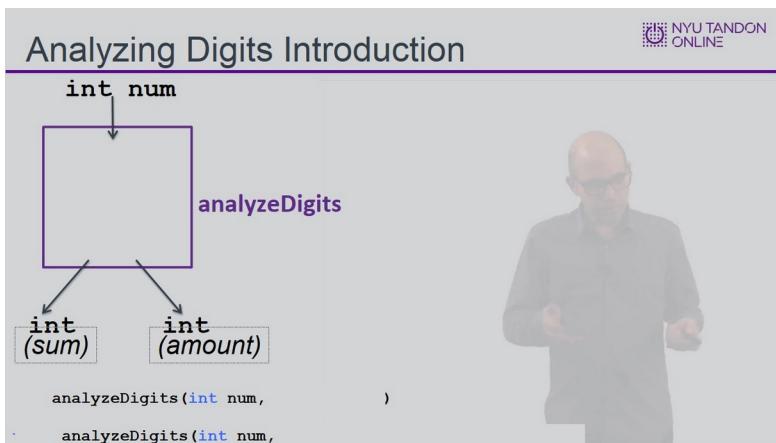
### Analyze Digits

Problem  
Write a program that reads a positive integer num, and prints the number of digits in num and their sum.

Example  
Please enter a positive integer:  
375  
375 has 3 digits and their sum is 15



### 3.5 Analyzing Digits Introduction



### 3.6 Analyze Digits Screencast

A screenshot of the Xcode IDE interface. The top menu bar includes File, Edit, View, Find, Navigate, Editor, Product, Debug, Source Control, Window, Help, and a search bar. The status bar at the bottom right shows "100% 600x1024 Fri Nov 18 12:14 Itay Tal". The main editor window displays the following C++ code:

```
#include <iostream>
using namespace std;
int main() {
    return 0;
}
```