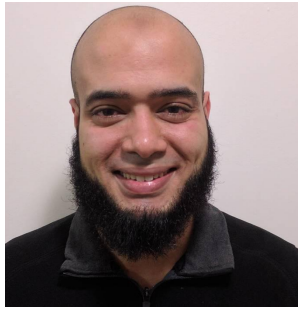
Programming 4 kids

Selection

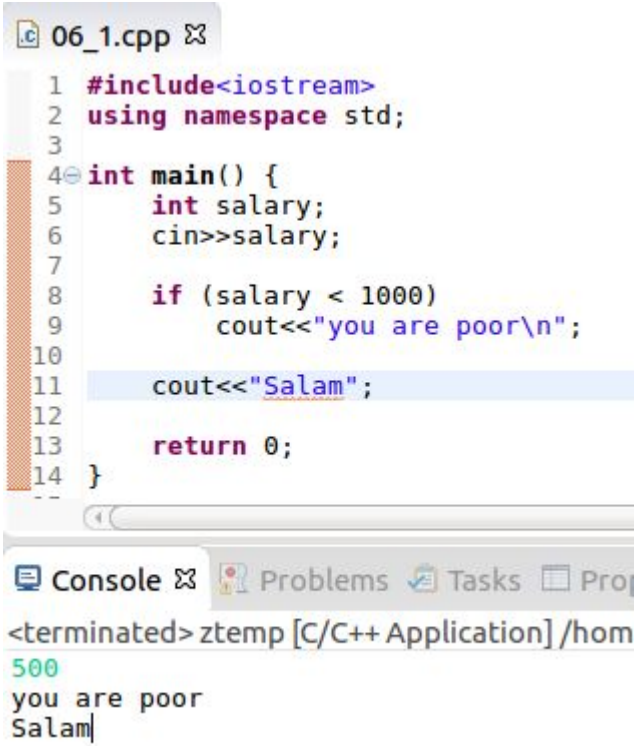
Mostafa Saad Ibrahim
Computer Vision Researcher @ Huawei Canada
PhD - Simon Fraser University
Bachelor / Msc - FCI Cairo University

Ex-(Software Engineer / Teaching Assistant)

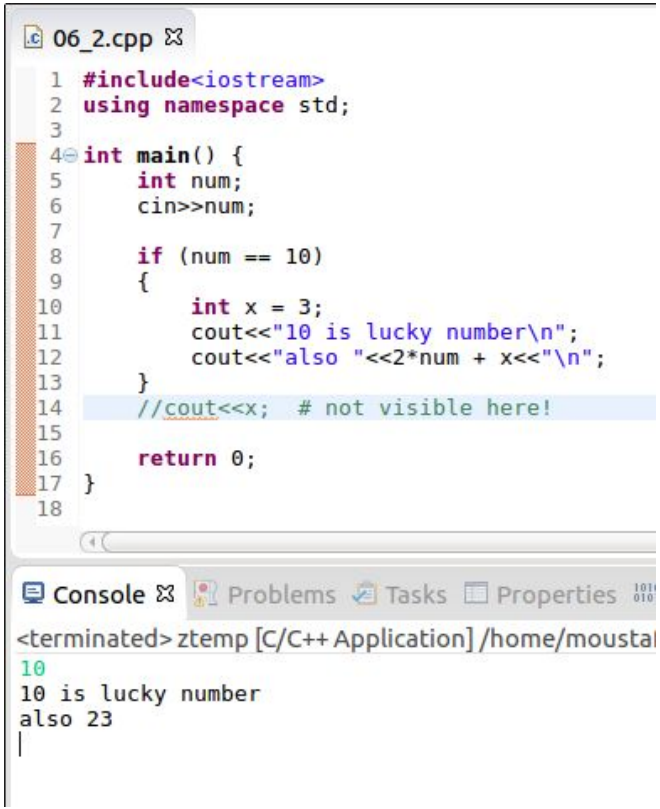


If conditions (الشروط)

- Write a program that reads an integer salary
- Then **If salary < 1000**,
  - print you are poor
- Otherwise do nothing
- End program with printing Salam
- …
- If (condition)
  - Body



What if I want big body?

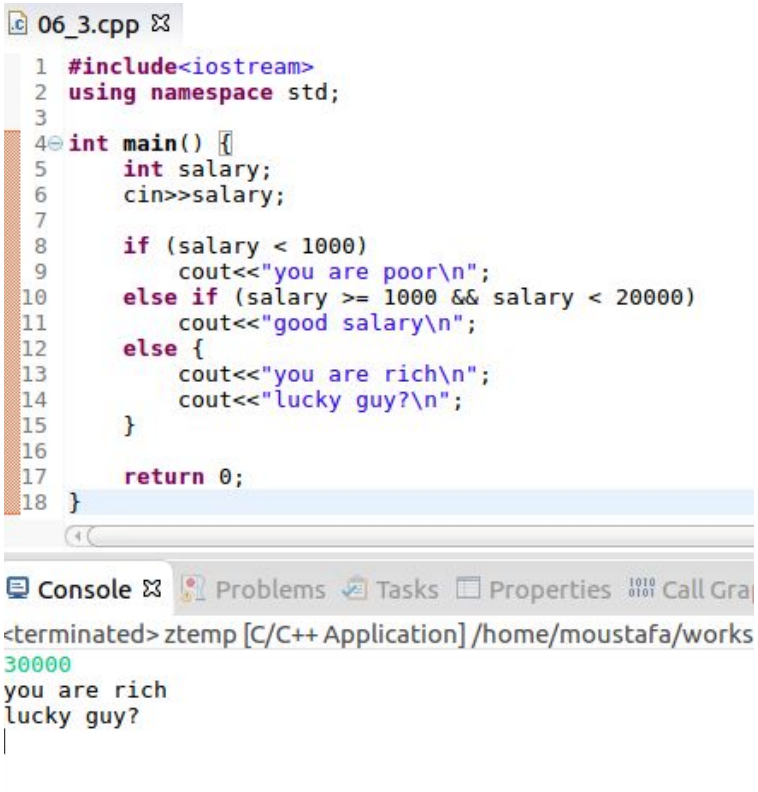


- Use { } if want to do more logic
  - You can write whatever
- Be careful, from scope
  - What inside {} not visible outside it

What if I need more conditions?

- Write a program that reads an integer salary then:
- If salary < 1000,
  - print you are poor
- Else If salary >= 1000 and < 20000,
  - print good salary
- Else If salary >= 20000,
  - print you are rich
- Now, how to command computer to do these if else?
- Good software engineer **tests** well his code. What are good **test cases**?
  - 0, 500, **1000**, **10000**, 200000, 100000000

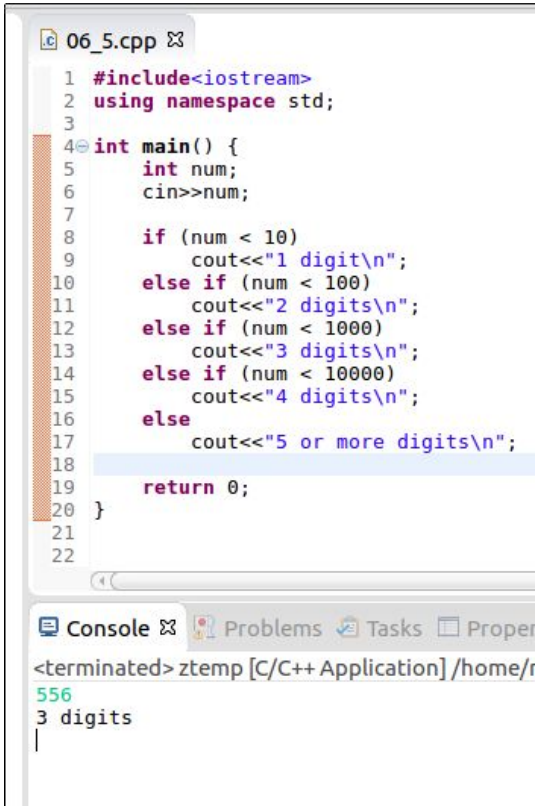
The if-else Chain



- Code follow
- If (cond)
  - Go to body ONLY if cond true
- If previous condition is not correct, move to next
- Else if (cond)
  - Again true, get it in
- If previous false move to next
- Else
  - Get in if nothing previous worked

How many digits?

- Read and integer and print if it has 1, 2, 3, 4 or 5+ digits
- For example if input is 556
  - Then print: **3 digits**



Nested if conditions

06_4.cpp

```
2  using namespace std;
3
4  int main() {
5      int salary;
6
7      cout<<"Enter salary: ";
8      cin>>salary;
9
10     if (salary < 1000)
11     {
12         cout<<"Enter age: ";
13         int age;
14         cin>>age;
15
16         if (age < 22)
17             cout<<"You are still young.";
18     }
19     else
20         cout<<"you are rich\n";
21
22     return 0;
23 }
24
25
```

Console | Problems | Tasks | Properties

```
<terminated> ztemp [C/C++ Application] /home/moustafa
Enter salary: 500
Enter age: 20
You are still young.
```

- Inside the body scope, we can do whatever
- Even another if (nested if)
- Or if { if {if () } }
- So whatever code body

Simple Calculator

- Given two **numbers** and a sign between them which will indicate if the user want the addition, subtraction, division or multiplication of these two numbers, find the value of the answer.
- Inputs ⇒ outputs
  - 7 + 55 ⇒ 62
  - 7 * 10 ⇒ 70
- **Stop** the video for a few minutes, and sketch some code

Simple Calculator

07_6.cpp

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

int main() {
    // Good choice for a number here is double
    double num1, num2;
    char operation;

    cin >> num1 >> operation >> num2;

    if (operation == '+')
        cout << num1 + num2 << "\n";

    else if (operation == '-')
        cout << num1 - num2 << "\n";

    else if (operation == '*')
        cout << num1 * num2 << "\n";

    else
        cout << num1 / num2 << "\n";

    return 0;
}
```

Console | Problems | Tasks | Properties | Call Graph

<terminated> ztemp [C/C++ Application] /home/moustafa/workspa

```
3 * 6
18
```

Minimum of 2 numbers

- Read 2 **integers** and print the minimum one of them
- Inputs ⇒ outputs
  - 10 20 ⇒ 10
  - 70 5 ⇒ 5
- Stop the video for a few minutes, and sketch some code

Minimum of 2 numbers

07_7.cpp

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

int main() {
    int num1, num2;

    cin >> num1 >> num2;

    if (num1 < num2)
        cout << num1 << "\n";
    else
        cout << num2 << "\n";

    return 0;
}
```

Console | Problems | Tasks | Propert

```
<terminated> ztemp [C/C++ Application] /home/m
20 10
10
```

Minimum of 3 numbers

- Read 3 **integers** and print the minimum one of them
- Inputs
  - 10 20 30 ⇒ 10
  - 70 5 15 ⇒ 5
- Stop the video for a few minutes:
  - Think what all cases setups we need to make sure code is correct?
  - Sketch the code. There are many ways to code it!

Min of 3 numbers: Way #1

07_8_A.cpp

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

int main() {
    int num1, num2, num3;

    cin >> num1 >> num2 >> num3;

    if (num1 < num2) {
        // Then either num1 or num3 is the answer
        if (num1 < num3)
            cout << num1 << "\n";
        else
            cout << num3 << "\n";
    } else  // Then either num2 or num3 is the answer
    {
        if (num2 < num3)
            cout << num2 << "\n";
        else
            cout << num3 << "\n";
    }

    return 0;
}
```

Console | Problems | Tasks | Properties | Call Graph

<terminated> ztemp [C/C++ Application] /home/moustafa/workspaces

```
20 10 30
10
```

Min of 3 numbers: Way #2

07_8_B.cpp

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

int main() {
    int num1, num2, num3;

    cin >> num1 >> num2 >> num3;

    if (num1 < num2 && num1 < num3)
        cout << num1 << "\n";
    else if (num2 < num1 && num2 < num3)
        cout << num2 << "\n";
    else
        cout << num3 << "\n";

    return 0;
}
```

Console | Problems | Tasks | Properties

```
<terminated> ztemp [C/C++ Application] /home/moustal
2 1 3
1
```

Min of 3 numbers: Way #3

07_8_C.cpp

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

int main() {
    int num1, num2, num3;

    cin >> num1 >> num2 >> num3;

    int answer = num1;

    if (answer > num2)
        answer = num2;

    if (answer > num3)
        answer = num3;

    cout << answer << "\n";

    return 0;
}

```

Console Problems Tasks Prop

<terminated> ztemp [C/C++ Application] /home

```
2 3 1
1
```

Summary: 3 styles

- If (condition)
  - body
- Body Either:
  - 1 line code OR
  - {

    Several lines

  - }
- Also body can be nested ifs

- If (condition)
  - body
- else if (condition)
  - body
- else if (condition)
  - body
- else if (condition)
  - body
- **else if** (condition)
  - body

- If (condition)
  - body
- else if (condition)
  - body
- else if (condition)
  - body
- else if (condition)
  - body
- **else**
  - body

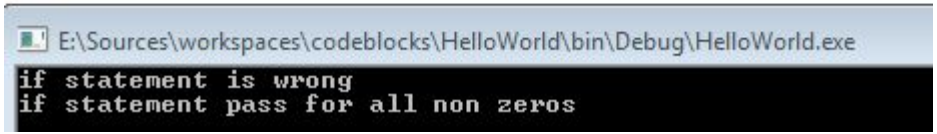
Be Careful

```
int age = 45;

if(age == 40)
    cout<<"Wow..40 years ago\n";

if(age = 40)
    cout<<"if statement is wrong\n";

if(age)
    cout<<"if statement pass for all non zeros\n";
```



- If (true condition)
- Be careful from these 2 mistakes
- Also remember: if no braces, write ONE line only

Practice: Is even? Print digits

- Read an integer N, then do the following
  - If the number is even: **print** last digit of it
  - If the number is odd: do following:
    - If number < 1000, **print** last 2 digits
    - If number >= 1000 and number < 1000000, **print** last 4 digits
    - Otherwise, **print** its negative value
- Stop the video and think: 1) Code 2) Good tests
- Testing examples of good coverage:
  - 234 ⇒ even ⇒ 4
  - 157 ⇒ 57
  - 567169 ⇒ 7169
  - 1000001 ⇒ -1000001

Practice: Is even? Print digits

06_6.cpp

```
3
4  int main() {
5      int num;
6      cin >> num;
7
8      bool is_even = (num % 2 == 0);
9
10     if (is_even)
11         cout << num % 10 << "\n";
12     else {
13         if (num < 1000)
14             cout << num % 100 << "\n";
15         else if (num < 1000000)
16             cout << num % 10000 << "\n";
17         else
18             cout << -num << "\n";
19     }
20     return 0;
21 }
22
23
```

Console | Problems | Tasks | Properties | Cal

```
<terminated> ztemp [C/C++ Application] /home/moustafa/w
567169
7169
```

- Recall %2 can be used to know if number is even
  - 0 ⇒ even
  - 1 ⇒ odd
- Notice we have if for even, then else for odd
- This else has big body for handling the 3 odd cases

Practice: Last 3 digits!

- Read an integer and do the following:
  - If number < 10000, **say** this is a small number
  - Otherwise Sum the last 3 digits of the number
    - If the sum is odd, **say** this is a great number
    - Otherwise, If sum is even:
      - If any digit of the last 3 is odd, **say** this is a good number
      - Otherwise, **say** this is a bad number
- Stop the video and think: 1) Code 2) Good tests
  - Be a good tester:
  - Find all needed test cases that covers all possible scenarios

Practice: Last 3 digits!

07_7.cpp

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

int main() {
    int n;
    cin>>n;

    if (n < 10000)
        cout<<"this is a small number\n";
    else
    {
        int digit1 = n%10;
        n = n/10;
        int digit2 = n%10;
        n = n/10;
        int digit3 = n%10;  // old value of n gone

        int sum = digit1+digit2+digit3;

        if ((sum%2) != 0)   // odd
            cout<<"this is a great number\n";
        else
        {
            bool is_digit1_odd = (digit1 % 2 == 1);
            bool is_digit2_odd = (digit2 % 2 == 1);
            bool is_digit3_odd = (digit3 % 2 == 1);

            if (is_digit1_odd || is_digit2_odd || is_digit3_odd)
                cout<<"this is a good number\n";
            else
                cout<<"this is a bad number\n";
        }
    }
    return 0;
}

```

- Test cases:
  - 100
  - 10111
  - 10330
  - 10303
  - 10033
  - 10000

Homework 1

- Read 2 integers A, B and print based on following cases:
  - if both are odd print their product A*B
  - if both are even print their division A/B
  - if the first is odd and the second is even then find their sum A+B
  - if the first is even and the second is odd then find their subtraction A-B
- Inputs ⇒ outputs
  - 5 7 => 35
  - 12 2 => 6
  - 5 6 => 11
  - 12 3 => 9

Homework 2: Sort 3 numbers

- Given 3 integers, sort (order) them in ascending order and print them .
- Inputs
  - 1 2 3 ⇒ 1 2 3
  - 1 3 2 ⇒ 1 2 3
  - 2 1 3 ⇒ 1 2 3
  - 2 3 1 ⇒ 1 2 3
  - 3 1 2 ⇒ 1 2 3
  - 3 2 1 ⇒ 1 2 3
- Do you notice there are only 6 ways to permutate 3 numbers!

Homework 3: Maximum but constrained

- Given 3 integers, you have to find the biggest one of them which is < 100.
  - Print -1 if no such number
- Inputs
  - 22 90 115 ⇒ 90
    - Here [20 90] are only < 100. Maximum (20, 90) = 90
  - 200 300 400 ⇒ -1
    - All of them are > 100, so no answer
  - 50 100 150 ⇒ 50
    - Only 50 is < 100.
  - 10 30 20 ⇒ 30
    - The 3 numbers < 100, so their max is 30

Homework 4: Conditional Count

- Write a program that reads number X, then other 5 numbers. Print 2 values:
  - How many numbers <= X
  - How many numbers > X
  - Any relation between these 2 outputs?
- Inputs
  - 10 **300 1 5 100 200**
  - Output: 2 3
  - Explantation
  - 2 numbers (1, 5) are <= 10
  - 3 numbers (100, 200, 300) are > 10

Homework 5: Find Maximum of 10

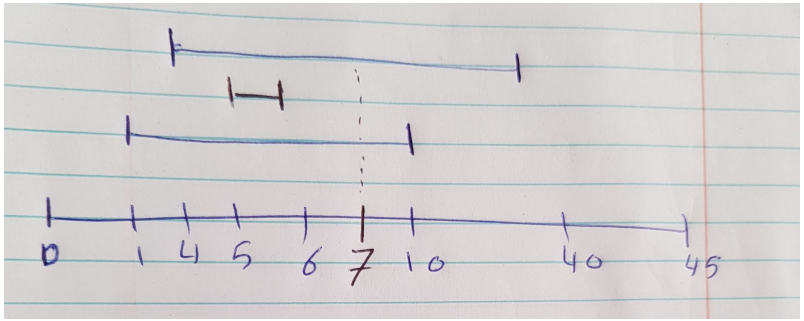
- Read 10 integers, find which of them has the biggest value and print it.
- Inputs
  - 1 67 -9 88 -45 129 90 65 77 34 ⇒ 129
- Restriction: In your whole code there should be 2 integer variables defined ONLY
  - If hard constraint; code it in whatever easier way for you

Homework 6: Find Maximum up to 10

- Read an integer N (2 <= N <= 10)
- Then read N integers, find which of them has the biggest value and print it.
- Inputs
  - 5 **1 3 2 4 2** ⇒ 4
    - 5 means read 5 integers
    - Then we read them [**1 3 2 4 2**]. Their maximum is 4
  - **10** 1 67 -9 88 -45 129 90 65 77 34 ⇒ 129
    - Same as last homework. This time we are given first N (10)
-

Homework 7: Intervals

- Read number X then read 6 numbers s1, e1, s2, e2, s3, e3
  - These 6 numbers are for 3 interval
  - Each Interval is a range [start, end]
  - Number X in a range if start <= X <= end
  - E.g 7 in range [5, 12] but not in range [10, 20]
- Print how many digits X is part of it
- Inputs
  - 7 1 10 5 6 4 40 ⇒ 2
    - Number 7 exists in 2 intervals [1, 10] and [4, 40]
  - 10 5 15 6 100 3 30 ⇒ 3
    - 1o exists in the 3 intervals [5 15], [6 100], [3 30]
  - 10 100 200 100 101 120 170 ⇒ 0



Homework 8: Two Intervals Intersection

- Read 4 numbers representing 2 intervals and print their intersection interval. If they don’t intersect, print -1
- Inputs
  - 1 6 3 8 ⇒ 3 6
    - Interval [1 6] and [3 8] only intersects at [3, 6]
    - Why: interval [1, 6] has numbers: {1, 2, **3, 4, 5, 6**}
    - And: interval [3, 8] has numbers: {**3, 4, 5, 6**, 7, 8}
    - So the intersection is {**3, 4, 5, 6**} = [3, 6]
  - 1 15 20 30 ⇒ -1

Finally

- Study well, then listen/do homeworks
  - Practice makes perfect
- Psychology:
  - It is ok to not be able to solve some homeworks by yourself
  - It is ok to not be able to understand homeworks solutions up to 50% of them
- Thinking & Coding
  - Long code ⇒ More bugs
  - Think deep ⇒ Code less
- Note: Some of the homeworks **during the course** may be from 2 sources:
  - AAST old archive / Assuit sheet

تم بحمد الله

علمكم الله ما ينفعكم

ونفعكم بما تعلمتم

وزادكم علماً