# DISCUSSION SESSION
# WEEK 3

# EXAM 01 REVIEW

# The Command Line

- ls: list the contents of the current directory
- cd: change the current directory
- pwd: print the current working directory
- mkdir: make a new directory
- rmdir: delete an empty directory
- rm –r: delete a non-empty directory
- touch: create a new file
- rm: delete a file
- cp: copy a file
- mv: move a file  cat: concatenate and display file contents

# Exercises

1. What command would you use to create a new directory called **project-1**? **Answer: mkdir project-1**
2. How would you move into the **project-1** directory?

   **Answer: cd project-1**
3. What command would you use to create a new file called **index.html**? **Answer: touch index.html**
4. How might you rename **index.html** to **home.html**?

   **Answer: mv index.html home.html**
5. What command would you use to list the contents of the current directory, including the hidden ones? **Answer: ls -a**

# Boolean Expressions (Single & Compound)

A **Boolean expression** is a specific kind of expression whose value when evaluated results in **true** or **false** (sometimes **1** or **0**, respectively).

```
bool myBooleanExpression = (x >= 25);
```

We can certainly combine two or more individual Boolean expressions into a single compound expression, using *logical operators*!

```
bool myExpression = (x > 0 && y < 10) || (z == 5 && w != 0);
```

```
bool myExpression = (x >=a && x <= b && y >= a && y <= b);
```

# Exercise

Write a Boolean expression that evaluates to true if a given integer **x** is either a multiple of 5 and not a multiple of 3, or it is a multiple of 4 and is between 10 and 50 (inclusive), and false otherwise.

```
bool exp = ((x % 5 == 0 && x % 3 != 0) || (x % 4 == 0 && x >= 10 && x <= 50))
```

# Number Systems

✓ **Decimal to Binary:** You can use the following algorithm

1. Divide the decimal number by 2
2. Get the integer quotient for the next iteration
3. Get the remainder for the binary digit
4. Repeat the steps until the quotient is equal to 0
5. The binary number is the remainder read from bottom to top

**Example:** To convert the decimal number 10 to binary:

10 / 2 = 5 remainder 0

5 / 2 = 2 remainder 1

2 / 2 = 1 remainder 0

1 / 2 = 0 remainder 1 **Therefore, the binary of 10 is 1010**

# Number Systems

- ✓ **Binary to Decimal:** To convert a binary number to decimal, you can use the following formula

$$d = (b_0 \times 2^0) + (b_1 \times 2^1) + (b_2 \times 2^2) + \ldots + (b_n \times 2^n)$$

..where **d** is the decimal number and $b_0, b_1, b_2, \ldots, b_n$ are the binary digits read backwards

# Number Systems

- ✓ **<u>Hexadecimal to Decimal:</u>** To convert a hexadecimal number to decimal, you can use the following formula

$$\mathbf{d = (h_0 \times 16^0) + (h_1 \times 16^1) + (h_2 \times 16^2) + \ldots + (h_n \times 16^n)}$$

..where **d** is the decimal number and $\mathbf{h_0, h_1, h_2, \ldots, h_n}$ are the hexadecimal digits read backwards

# Number Systems

✓**Decimal to Hexadecimal:** You can use the following algorithm

1. Divide the decimal number by 16
2. Get the integer quotient for the next iteration
3. Get the remainder for the hexadecimal digit. If the remainder is greater than 9, use the corresponding letter (A, B, C, D, E, F)
4. Repeat the steps until the quotient is equal to 0
5. The hexadecimal number is the remainder read from bottom to top

**Example:** To convert the decimal number 200 to hexadecimal:

200 / 16 = 12 remainder 8

12 / 16 = 0 remainder 12

**Therefore, the hexadecimal of 200 is C8**

# Exercises

1. Convert the decimal number **42** to a binary number.

**Answer: 101010**

2. Convert the binary number **101101** to a decimal number.

**Answer: 45**

3. Convert the decimal number **255** to a hexadecimal number.

**Answer: FF**

4. Convert the hexadecimal number **1A3** to a decimal number.

**Answer: 419**

# True or False?

1. C++ programs may not include a main function. **False.**
2. The *std::cout* object is used to read user input. **False.**
3. *break* can be used to exit a function prematurely. **False.**
4. The *default* case in a switch statement is mandatory in C++. **False.**
5. In C++, you can declare a function inside another function. **False.**
6. Predefined functions can be invoked after including the library header. **True.**

# True or False?

```
include <iostream>
int main( ) {
    int x = 5;
    int y = 0;
    std::cout << x / y << std::endl;
}
```

**T or F:** The above code will ***compile*** successfully.

**Answer: True, division by zero is a runtime error that the compiler won't catch at compile-time.**

# Expected Output?

```
include <iostream>
int main( ) {
    int x = 3;
    do {
        std::cout << "Exam Prep" << std::endl;
        x++;
    } while (x < 3);
}
```

How many times will **"Exam Prep"** print?

**Answer: Once, because do-while loop executes code block at least once before checking condition.**

# What's the Output?

```
#include <iostream>
int func(int x, int y) {
        int temp = x;
        x = y;
        y = temp;
        return temp - y;
}
int main( ) {
        int x = 5, y = 2;
        x = func(y, x);
        std::cout << func(x, y);
}
```

**Answer: 0**

# What's the Output?

```
#include <iostream>
int main( ) {
        int a = 5;
        int b = 10;
        if (a > b) {
                if (a > 0) {
                        std::cout << "A is greater than B and positive";
                }
        else {
                std::cout << "B is greater than A";
        }
        }
}
```

**Answer: No output**

# What's the Output?

Answer: 21

```
#include <iostream>
int func1(int x) {
    return x + 5;
}
int func2(int y) {
    y *= 2;
    return func1(y + 3);
}
int main( ) {
    int x = 3, y = 2;
    std::cout << func2(x) + func1(y);
}
```

# What's the Output?

```
#include <iostream>
int main( ) {
        int i = 3, j = 4;
        int k = ++i   +   ++j;
        k += i++   +   j++;
        i = k++   +   ++j;
        std::cout << i << " " << j << " " << k;
 }
```

Answer: 25  7  19

# What does this mystery function do?

```
#include <iostream>
int mysteryFunction(int x) {
    int result = 0;
    while(x > 0) {
        result += x % 10;
        x /= 10;
    }
    return result;
}
```

**Answer: Sums up all digits of a number**

# What does this mystery function do?

```
#include <iostream>
int mysteryFunction(int x) {
        int result = 1;
        for(int i= 2; i <= x; i++) {
                result *= i;
         }
         return result;
}
```

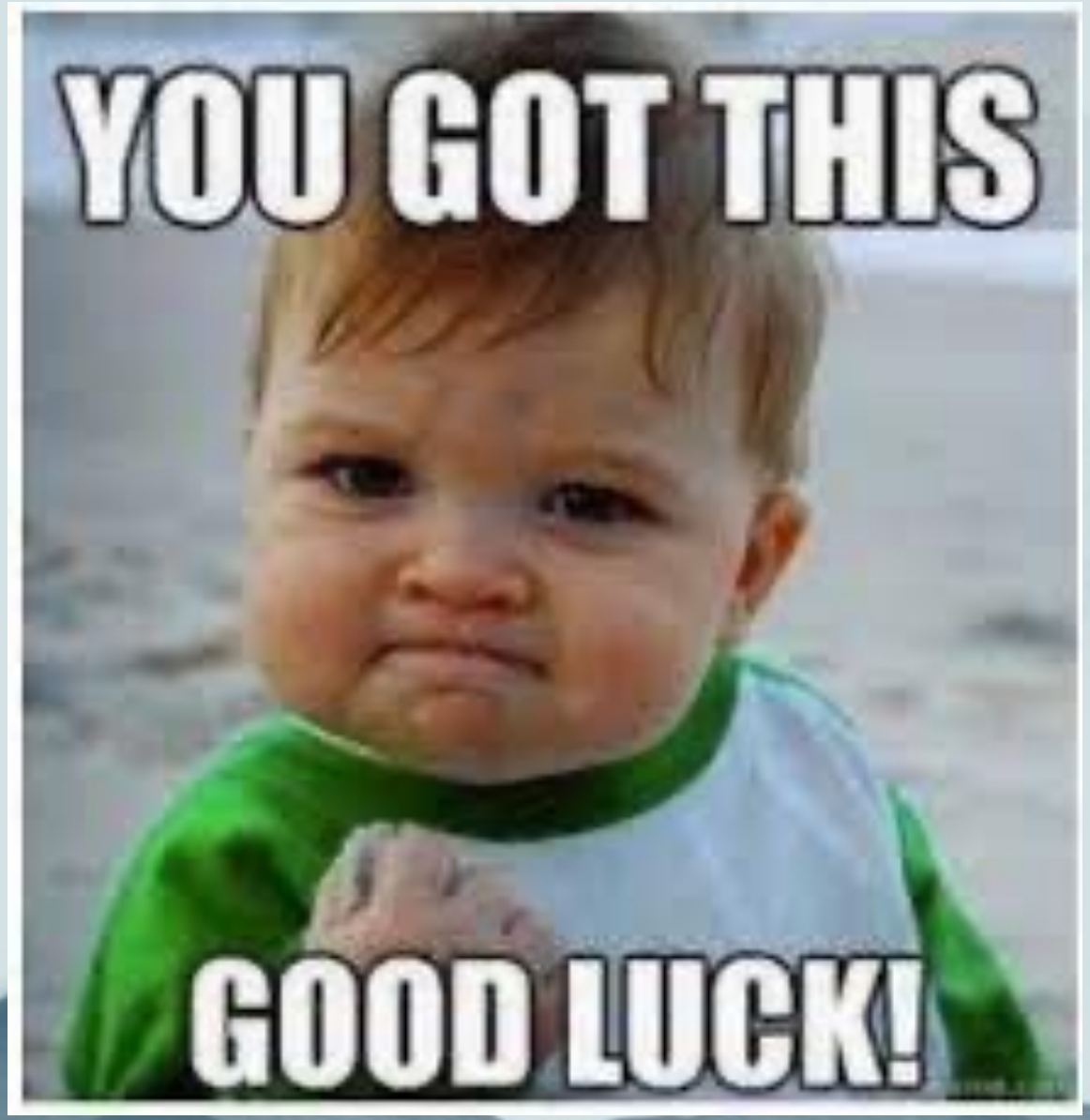
**Answer: Calculates factorial of an integer**

# Question?

- Write a function `highest_prime` that takes an integer `n > 1` from `stdin` and outputs the largest prime number less than or equal than `n` to the `stdout`
  - use an `is_prime` function to help find `highest_prime`

```cpp
int highest_prime(int n) {
    for (int i = n; i > 1; i--) {
        bool isPrime = true;
        for (int j = 2; j <= i/2; j++) {
            if (i % j == 0) {
                isPrime = false;
                break;
            }
        }
        if (isPrime) return i;
    }
}

int main() {
    std::cout << highest_prime(32);
    return 0;
}
```


YOU GOT THIS
GOOD LUCK!