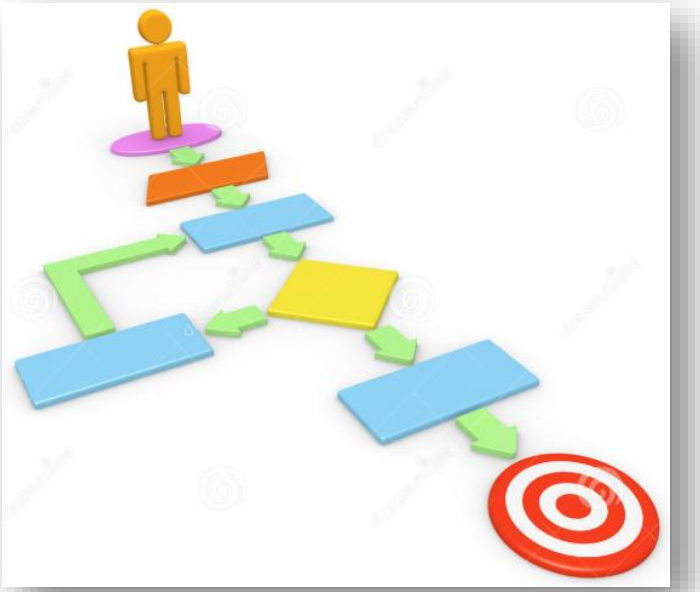
DATA STRUCTURE & PROGRAMMING I

Topic 1: Introduction

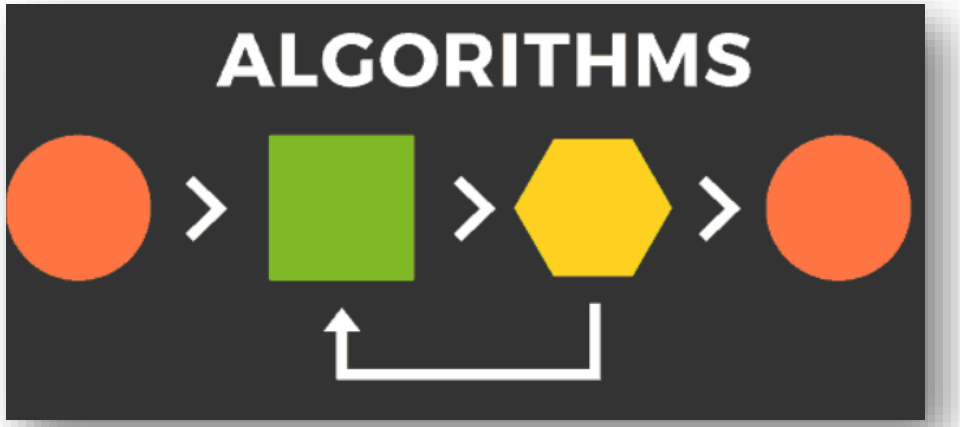




Algorithm

❑ What is algorithm?

- An algorithm is a **step-by-step procedure** describing how to solve a problem / reach a goal
- It may take some ***input***, *process the input* to achieve the goal, and finally display the *output/result* if needed



- **Example**:
  - How to make a fruit salad
  - How to validate an email address
  - How to add two numbers from a user's inputs

Algorithm Examples (1/3)

❑ Example 1: How to make a "Fruit Salad"

Ingredient (input)

Procedures

Product (output)

```
You will need:
2 bananas
2 strawberries
2 oranges
2 apple

What to do:
1. Wash your hands and clean your cooking area
2. Wash bananas, strawberries, oranges, apples
3. Peel bananas and oranges
4. Cut all the fruits according to recipe
5. Place fruit in a large bowl and mix

Output:
Fruit salad
```

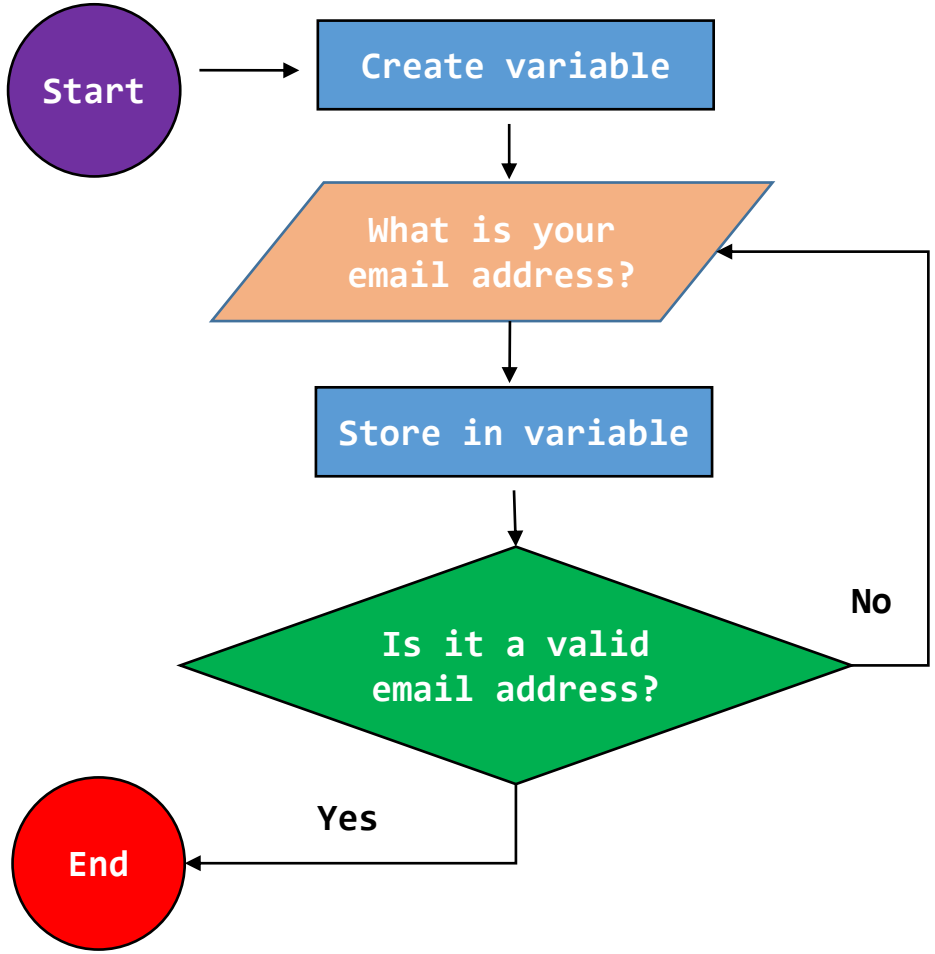
Algorithm Examples (2/3)

❑ Example 2: How to validate an input email address

START

- Declare variable *address* (text)
- Ask user to input address
- Do operation: check address if valid
  - If not valid, user input again
  - If valid, done.

STOP



Algorithm Examples (3/3)

❑ Example 3: How to add two numbers (more specific to coding)

START

- Declare variable a, b, c (Number)
- Get values a and b from user
- Do operation c = a +b
- Display c

STOP

Remarks

❑ Algorithm and Computer Programming Language

- An algorithm is not computer code
  - It's just written in simple English or whatever the programmer speaks
  - But it has a *start*, a *middle* and an *end*

- Converting algorithm to a specific **computer programming language** is needed to make computer understand and solve a given problem
  - **C, C++, C#, Java, PHP, JavaScript etc.**

Algorithm

❑ Why algorithm?

- ✓ It gives us idea how to deal with a problem
- ✓ It helps us to analyze a problem
- ✓ It also tells us how to solve a problem
- ✓ Generally, a problem can be solved by more than one algorithm

- Which algorithm is the best to solve a given problem?
  - Time consuming
  - Memory usage
  - Code Length of applicable programming language (how long is the applicable code)
  - Complexity

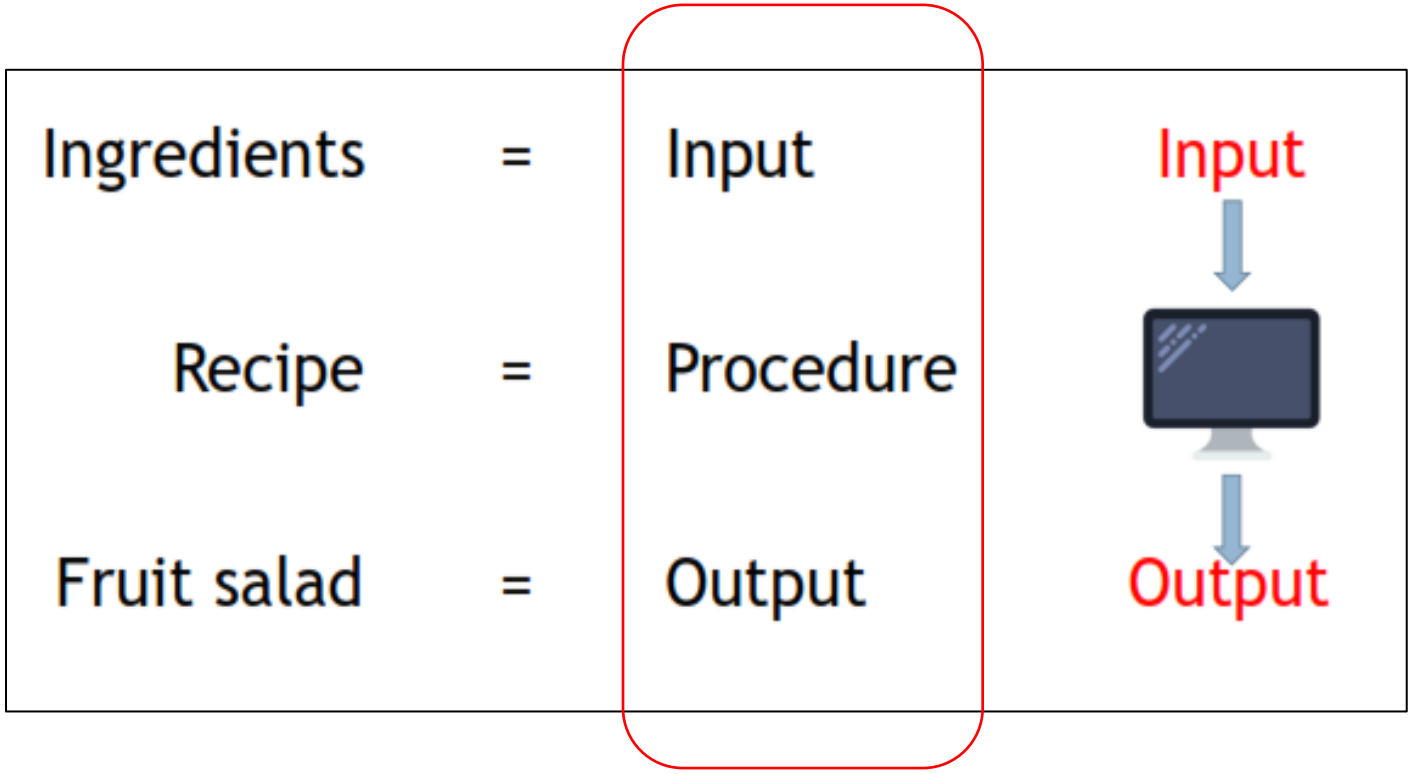
Algorithm

❑ Fundamental idea on making an algorithm?

- When a problem is given,
  - Check it whether there is an algorithm can solve it
  - Then how fast or complicated is your algorithm can solve the problem
  - Check the result whether it is what you need

Algorithm

❑ Main components of algorithm

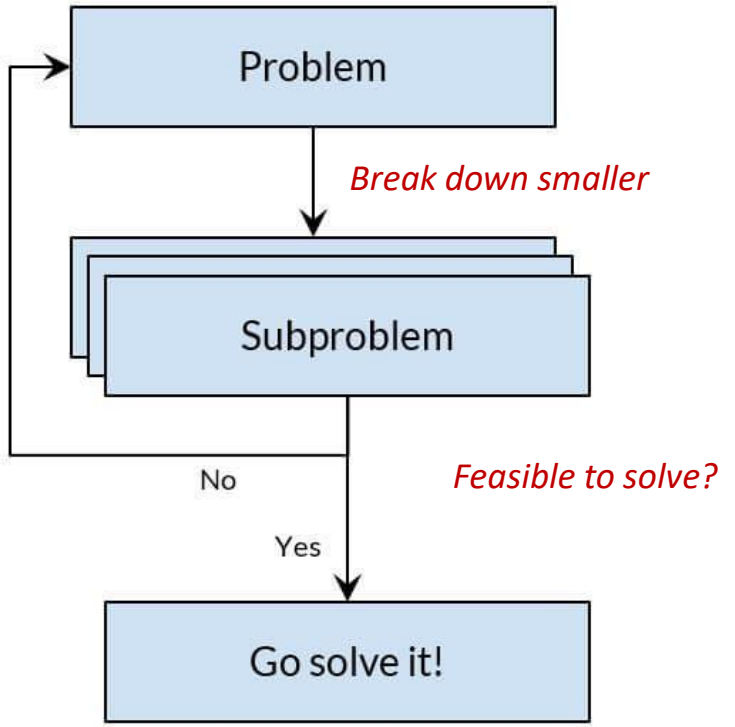


Referred to 1st example on how to make a fruit salad

Algorithm

❑ How to write an algorithm?

1. Divide a given problem into sub-problems
2. Arrange sub-problem in order
3. Determine the solution for each sub-problem
4. Solve each sub-problem
5. Then, the whole problem is solved



Algorithm

❑ Important guidelines

- Clarity
  - Semantic
    - Meaning of program
  - Syntactic
    - Grammar/rule/formula/syntax written in program

- Readability
  - Separation of different information
    - Declaration
    - Calculation or process
    - Displaying information
  - Indentation
    - Use tab when needed

Algorithm

❑ Structure of writing an algorithm

- Consist of two main parts
  1. Variable declaration (reserve memory space)
  2. Algorithm

- **Get Input,**
- **Process input,**
- **Output result**

```
Var name1, name2 : Type of variable
Begin
     // your algorithms start here
      statement 1
      statement 2
      statement 3
End
```

An example of structure of an algorithm

Algorithm

❑ Example #1 (Simple)

```
Var n1, n2, result : Number
Begin
     Get n1, n2 from user
     Do operation result = n1+n2
     Display result

End
```

Sum two input numbers and display the result

Algorithm

❑ Example #2

```
Var n1, n2, result : Number
Begin
    read(n1, n2)
    result ← (n1+n2)*2
    write(result)

End
```

Multiply the sum of two input numbers with 2 and display the result

Algorithm

❑ Example #3

```
Var n1, n2 : Number
Begin
     read(n1, n2)
     n1 ← (n1+n2)*2
     write(n1)

End
```

Multiply the sum of two input numbers with 2 and store in the 1st number variable

Practice

1. Write an algorithm to greet message. Ask a user for a name then display welcome message

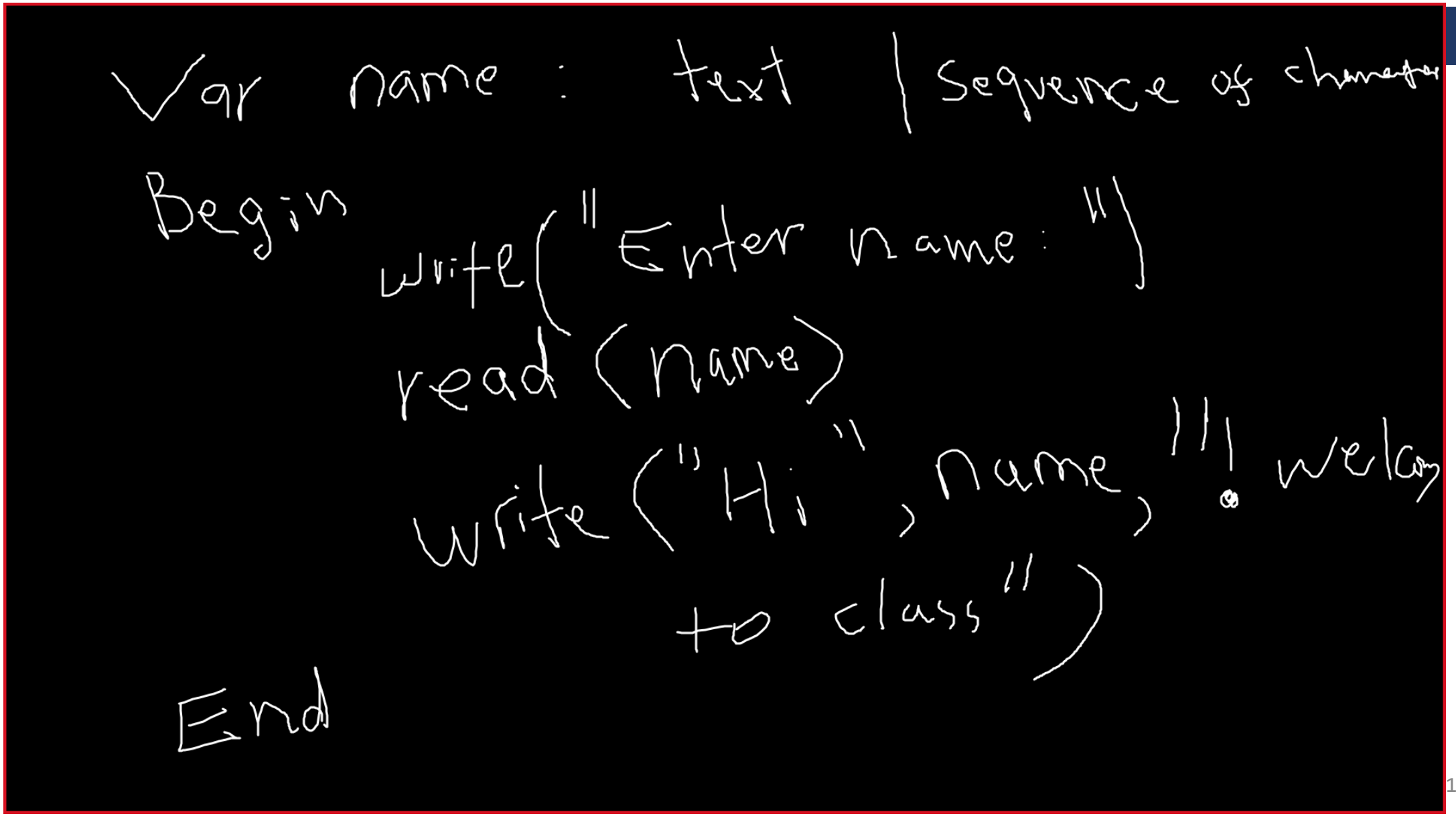
   Input name: John
   Hi, John! Welcome to our department!

2. Write an algorithm to ask year of birth of a user. Tell age of the user based on his/her given year of birth.

   Input year of birth: 2000
   You were born in 2000 and you are 21 year old.

3. Write an algorithm to do basic math operations + - * /. Get two input numbers from user then do the operation above.

#1


Var name : text | Sequence of chara
Begin
write("Enter name: ")
read(name)
write("Hi ", name, "!! welco
to class")
End

#2

```
Var y, age : Number

Begin
    Write("Input year birth: ")
    read(y)
    age ← 2021 - y
    Write("You were born in", y, ". You are", age, "year old")
End
```

C Programming



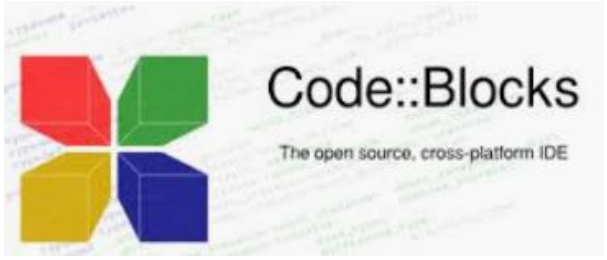
❑ Brief information

- First basic programming language
- Invented by Dennis Ritchie…….
- Mostly used to interact with machine (fast, secure, …)
- Short history
  - C was invented to write an operating system called UNIX.
  - C is a successor of B language which was introduced around the early 1970s.
  - The language was formalized in 1988 by the American National Standard Institute (ANSI).
  - The UNIX OS was totally written in C.
  - Most of the state-of-the-art software have been implemented using C.

C Programming

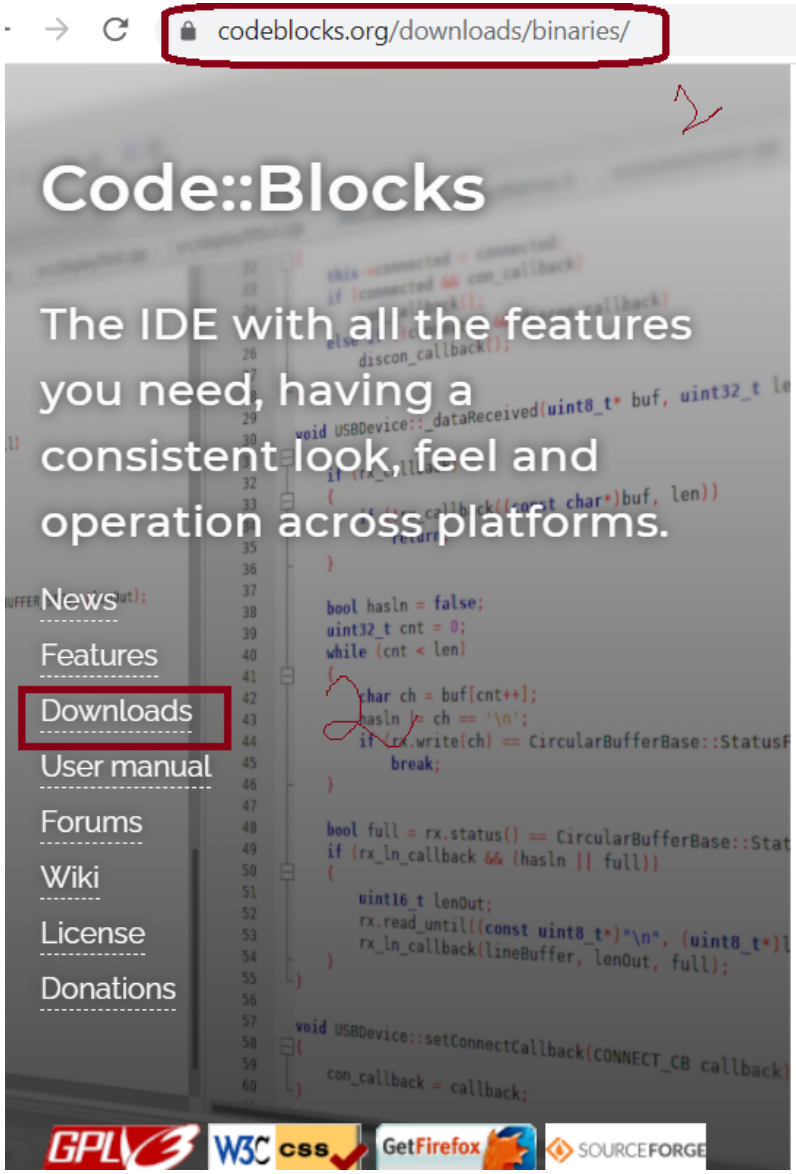
❑ Code syntax

- An extension of C programming is: **.c**
- Before starting out in C programming, make sure that you have a compiler
- A **compiler** turns code that you write into an **executable file** that computer can understand and run
- **Integrated Development Environment (IDE)** is a software application that provides comprehensive facilities (code color, code completion, …) to computer programmers for software development
  - **Code::Blocks** will be used



Download software and install: Code blocks

https://www.codeblocks.org/downloads/binaries/
Or use this short link: bit.ly/3WW72am



Code::Blocks
The open source, cross-platform IDE

Sourceforge.net page.

NOTE: There are also more recent nightly builds available in the **forums** or (for Ubu
PPA repository. Please note that we consider nightly builds to be stable, usually.

NOTE: We have a **Changelog for 20.03**, that gives you an overview over the enhance
put in the new release.

NOTE: The default builds are 64 bit (starting with release 20.03). We also p
convenience.

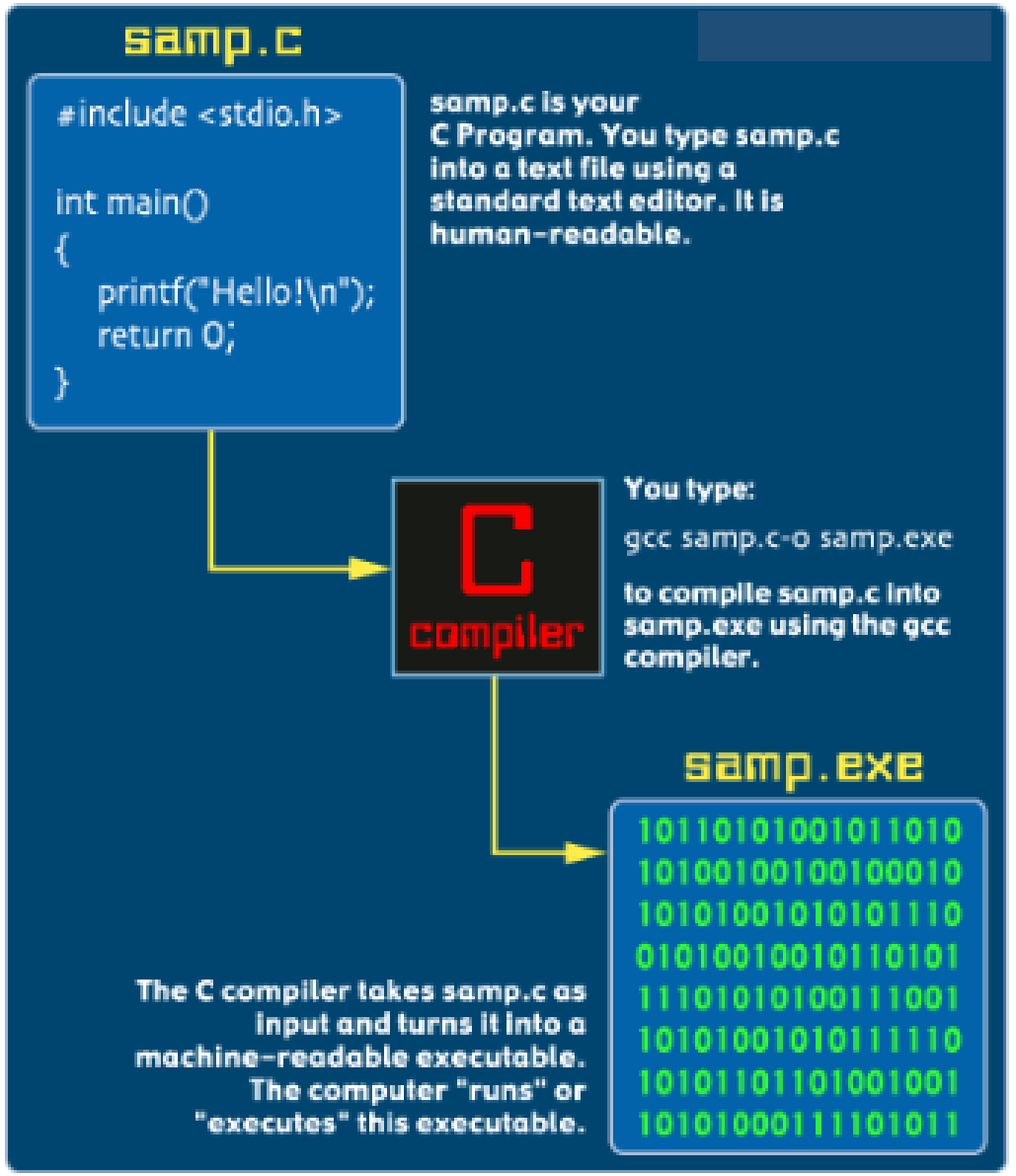
Microsoft Windows

File	Download from
codeblocks-20.03-setup.exe	FossHUB or Sourceforge.net
codeblocks-20.03-setup-nonadmin.exe	FossHUB or Sourceforge.net
codeblocks-20.03-nosetup.zip	FossHUB or Sourceforge.net
codeblocks-20.03mingw-setup.exe	FossHUB or Sourceforge.net
codeblocks-20.03mingw-nosetup.zip	FossHUB or Sourceforge.net
codeblocks-20.03-32bit-setup.exe	FossHUB or Sourceforge.net
codeblocks-20.03-32bit-setup-nonadmin.exe	FossHUB or Sourceforge.net
codeblocks-20.03-32bit-nosetup.zip	FossHUB or Sourceforge.net
codeblocks-20.03mingw-32bit-setup.exe	FossHUB or Sourceforge.net

C Programming

❑ Compiling C language

- Source code
  - `filename.c`
- How to compile code with command prompt **cmd**
  - `gcc filename.c –o filename.exe`
- Executable file
  - `filename.exe`



Structure of C programming

❑ Structure

- Code

```
#include <stdio.h>

main(){

      write code here
      ……

}
```

Header (library)
stdio: standard input output

main function
-It runs its codes inside when the program executes
-Use { } and put code inside it

Code
write your code here

C Programming

❑ Code syntax

- All C program statement begins inside a function called main()
- The main function is always called when the program first executes
- To access the standard functions that comes with compiler, a header with the
  - #include <stdio.h> need to be included on top

- Structure of code

```
#include<stdio.h>
main(){
    … Write your code here …
}
```

```
#include<stdio.h>
main(){
        printf("Hello World!");
        printf("Welcome to C Programming.");
}
```

Example of our 1st Program in C programming

- Each line of code ends with a semicolon ;

❑ Commands in C

- Creating variable
  - ✓ int a, b, c; : Integer
  - ✓ char ch; : Character
  - ✓ float price; : Floating number
  - ✓ char name[10]; : String (sequence of character)
- Display variable value
  - It requires placeholder
    - %d: for int
    - %f: for float
    - %c: for char
    - %s: for string (sequence of character)
- **Note**: To insert special character, use the following inside double quote “ ”
  - \n: for newline
  - \t: for tab

printf(“ asdfsadfasd ”);

printf(“asdfasdf %d”, variableName)

test1.c

```c
#include <stdio.h>
int main(){
    int age=22;
    char sex='M';
    char name[]="Dara";
    printf("Welcome new student: \n");
    printf("Name: %s\n", name);
    printf("Sex: %c\n", sex);
    printf("Age: %d\n", age);
}
```

```
Welcome new student:
Name: Dara
Sex: M
Age: 22
```

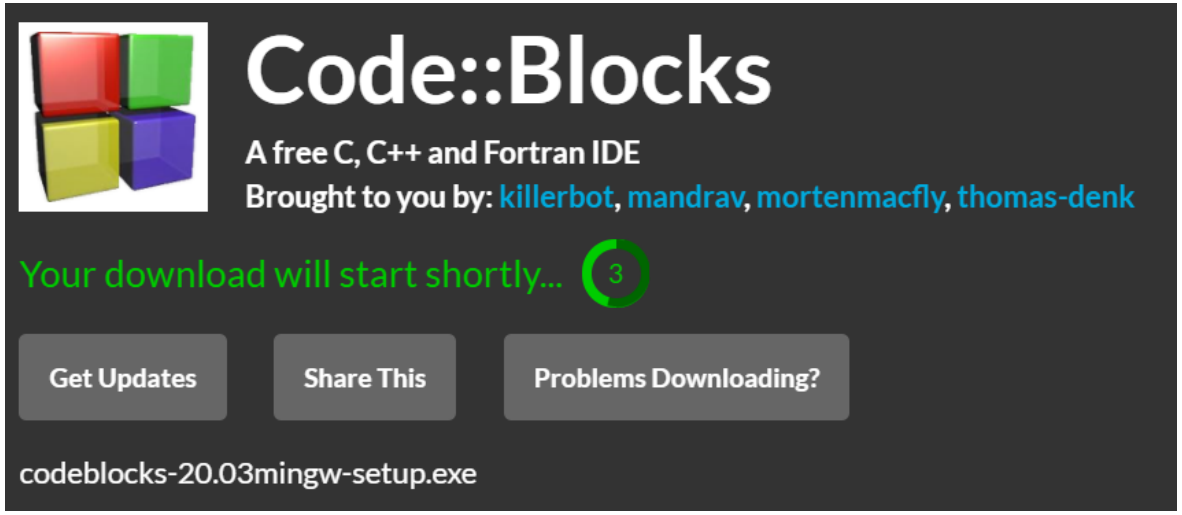
Q&A

Assignment

Deadline duration: **1 week**

❑ Task 1

- Download and install a program
  - **CodeBlock IDE**
  - Go to the following link and download
    - https://sourceforge.net/projects/codeblocks/files/Binaries/20.03/Windows/codeblocks-20.03mingw-setup.exe/download



Assignment

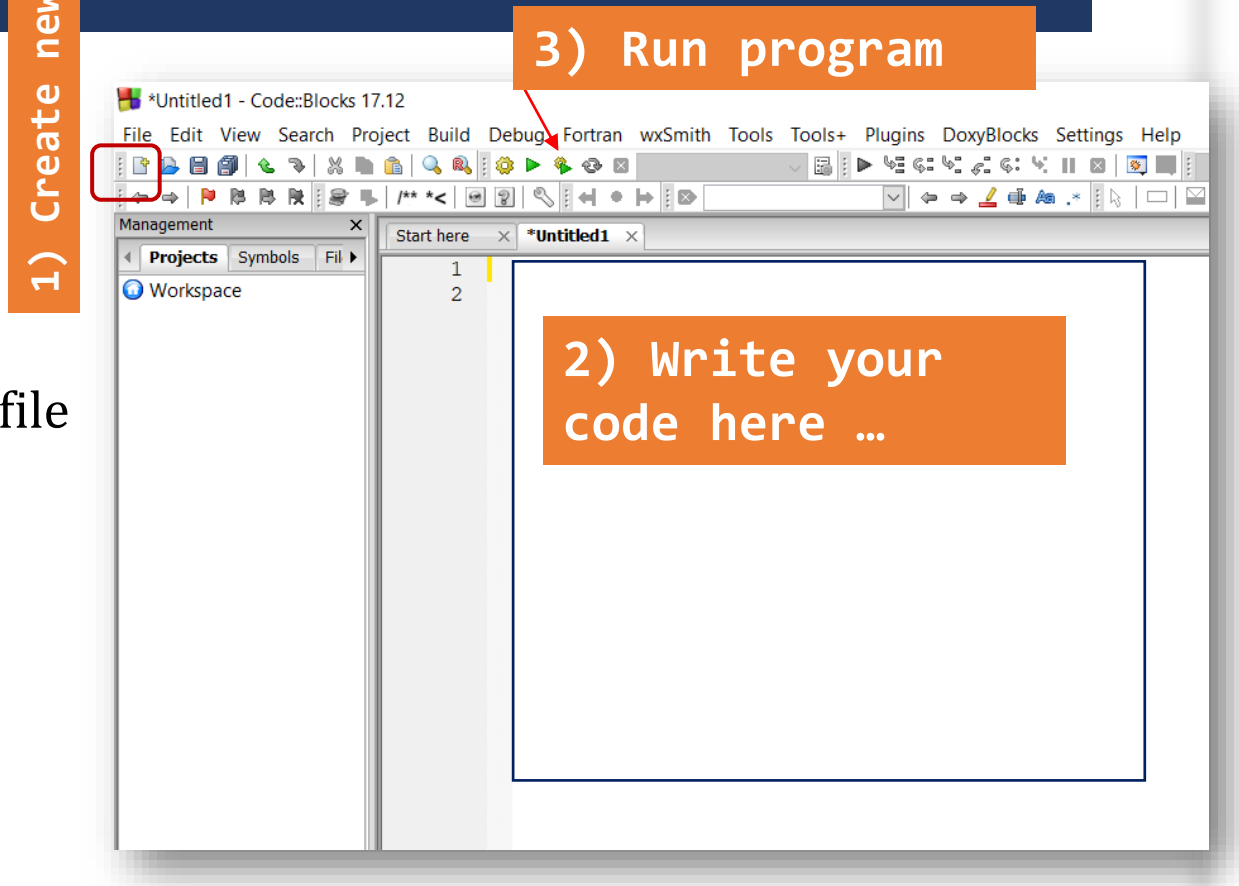
Deadline duration: **1 week**

❑ Task 2

- Create your first program
  - Open the installed program
  - **File -> new -> empty file**
    - Or just click on new file icon
  - Copy code below and paste to the created file
  - Save file: ctrl+s
  - Type name of file follow by **.c**
    - Example : **Test.c**

```
#include <stdio.h>
main(){
        printf(“"Hello World!")
        printf("Welcome to C Programming.")
}
```

Example of code in C programming



Interface of CodeBlock IDE