COE 251

INTRODUCTION TO C PROGRAMMING

Dr. Eliel Keelson

OUTLINE

WHAT IS C?

The Concept of Computer Programming

WHY LEARN C?

What will you gain if you Learn C?

COURSE OUTLINE

Topics to be covered

1

WHAT IS C?

The Concept of Programming

CHARACTERISTICS OF A COMPUTER

- All digital computers are basically electronic devices that can transmit, store and manipulate information (i.e., data).
- Several types of data can be processed by a computer.
- To process a particular set of data, the computer must be given an appropriate set of instructions called a **program**.
- These instructions are entered into the computer and then stored in a portion of the computer's memory

CHARACTERISTICS OF A COMPUTER

- A stored program can be executed at any time. This causes the following things to happen:

1. A set of information, called the **input data**, will be entered into the computer (from the keyboard, etc.) and stored in a portion of the computer's memory.
2. The input data will be processed to produce certain results, known as the **output data**.
3. The output data and perhaps some of the input data, will be printed on a sheet of paper or displayed on a monitor.

COMPUTER PROGRAMMING

- Computer programming can be described as the art of telling a computer what to do through a set of instructions.
- These written set of instructions are known as **Computer Code** or (just) **Code**
- So programming is also known as coding
- To Code you need an appropriate programming language

PROGRAMMING LANGUAGES

- A programming language is a set of commands with a governing syntax which is appropriate for instructing a computer or computing device to perform specific tasks.
- Programming languages can be used to create programs that implement specific **algorithms**.
- Algorithms are a **finite** sequence of steps which are followed in solving a particular problem

TYPES OF PROGRAMMING LANGUAGES

- There are basically two categories of programming languages. These are:

1. Low Level Languages which comprise **Machine** and **Assembly Languages**
2. High-Level Languages

TYPES OF PROGRAMMING LANGUAGES

- The machine code/language, contains a series of **binary** codes that are understood directly by a computer's CPU.
- Needless to say, machine language is not designed to be human readable.
- An assembly language contains a list of basic instructions and is much more difficult to read than a high-level language.

TYPES OF PROGRAMMING LANGUAGES

- Usually, a computer program will be written in some high-level language, whose instruction set is more compatible with human languages and human thought processes.
- High-level languages are designed to be easy to read and understand. This allows programmers to write **source code** in a natural fashion, using logical words and symbols.

TYPES OF PROGRAMMING LANGUAGES

- For example, reserved words like function, while, if, and else are used in most major high-level programming languages.
- Symbols like **<**, **>**, **==**, and **!=** are common operators.
- Many high-level languages are similar enough that programmers can easily understand source code written in multiple languages.

TYPES OF PROGRAMMING LANGUAGES

- Examples of high-level languages include C, C++, Java, Perl, Python, PHP and Ruby.
- A program written in high-level language must be translated into machine language before it can be executed.
- This is known as compilation or interpretation, depending on how it is carried out.

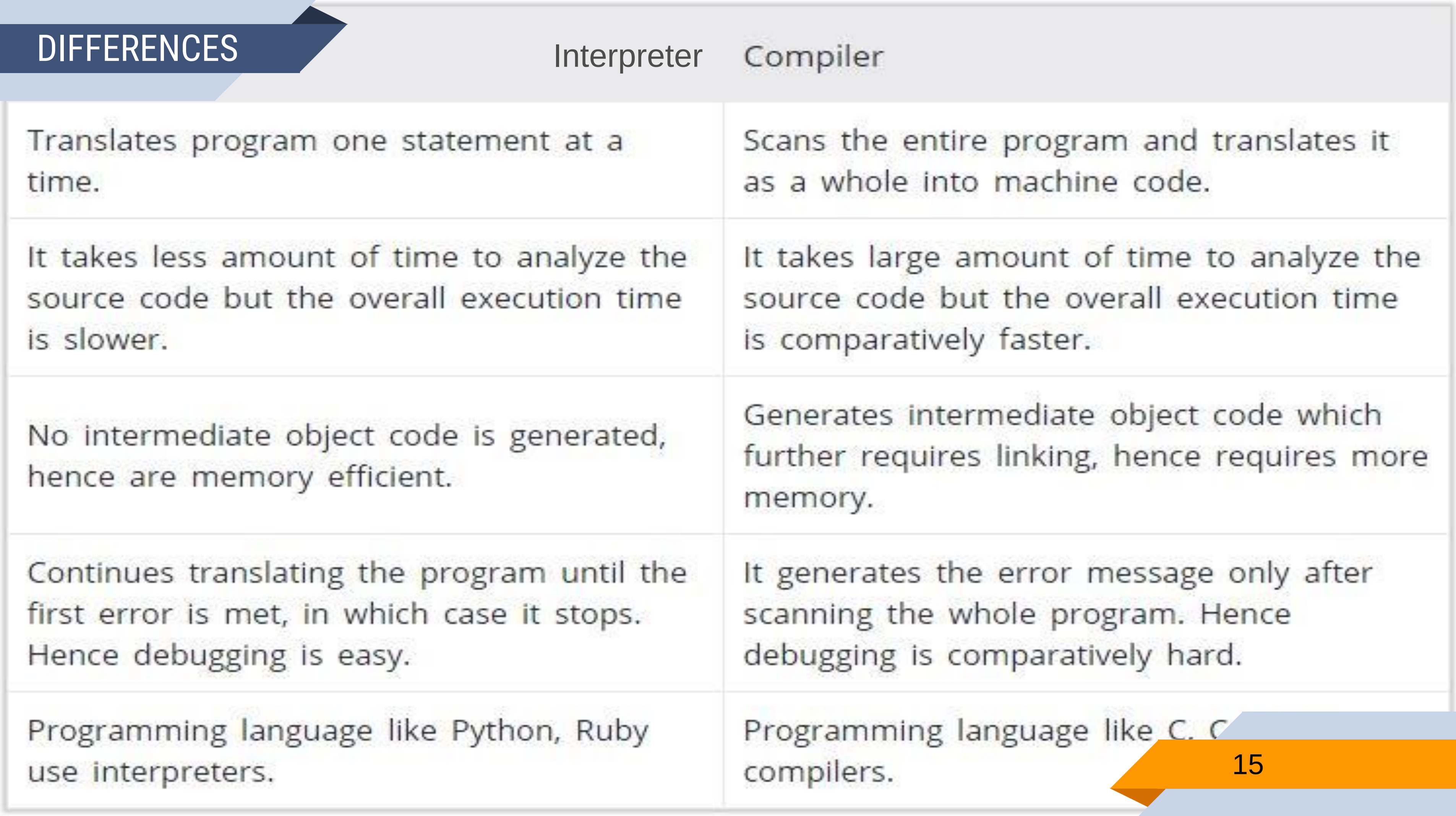
TYPES OF PROGRAMMING LANGUAGES

- An **Interpreter** is a program that translates programming language instructions one line at a time.
- A **Compiler** works by translating the entire program at one time.
- Languages like C, C++ and Java are called "**compiled languages**" since the source code must first be compiled in order to run.

TYPES OF PROGRAMMING LANGUAGES

- Languages like Perl, Python, PHP and Ruby are called "**interpreted languages**" since the source code can be run through an interpreter without being compiled.
- Generally, compiled languages are used to create software applications, while interpreted languages are used for running **scripts**, such as those used to generate content for **dynamic websites**.

DIFFERENCES



Interpreter	Compiler
Translates program one statement at a time.	Scans the entire program and translates it as a whole into machine code.
It takes less amount of time to analyze the source code but the overall execution time is slower.	It takes large amount of time to analyze the source code but the overall execution time is comparatively faster.
No intermediate object code is generated, hence are memory efficient.	Generates intermediate object code which further requires linking, hence requires more memory.
Continues translating the program until the first error is met, in which case it stops. Hence debugging is easy.	It generates the error message only after scanning the whole program. Hence debugging is comparatively hard.
Programming language like Python, Ruby use interpreters.	Programming language like C, C compilers.

WHAT IS C?

- C is a general-purpose high-level programming language used for wide range of applications; from Operating systems like Windows and iOS to software that is used for creating 3D movies.
- C programming is highly efficient. That’s the main reason why it’s very popular despite being more than 40 years old.

WHAT IS C?

- Standard C programs are portable; that is to say that, source code written in one system works in another operating system without any change.
- C is a good choice for programming newbies in starting their programming journey.
- When one knows C programming, he will not just understand how a program works, but will also be able to create a mental picture on how a computer works.

HISTORY OF C PROGRAMMING

- C is closely associated with Unix Operating system
- The developers of Unix Operating system (including Dennis Ritchie and Stephen C. Johnson) decided to rewrite the system in B language.
- However, B couldn’t suffice some of the features of that PDP-11 version of Unix, which led to the development of C.

HISTORY OF C PROGRAMMING

- In 1972, the development of C started on the PDP-11 Unix system.
- A large part of Unix was then rewritten in C.
- By 1973, C was powerful enough to be used in Unix Kernel.
- Dennis Ritchie and Stephen C. Johnson made further changes to the language for several years to make it portable in Unix Operating system.

2

WHY LEARN C?

What would you gain if you Learn C?

WHY LEARN C?

- Learning C makes the inner workings of a computer more understandable thus providing a clear mental model of how the computer works.
- C is the lingua franca of programming. Almost all high-level programming languages like Java, Python, JavaScript etc. can interface with C programming.

WHY LEARN C?

- Learning C provides one with the opportunity and capability to work on open source projects that impact millions of people.
- This is so because most high-level programming languages (e.g. Python) were birthed from C. Thus, C is the main backbone on which they ride.
- Therefore to contribute to how these other programming languages work, one must learn C.

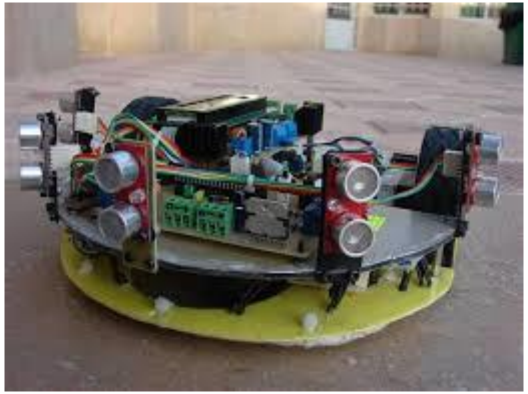
WHY LEARN C?

- Most programming languages use a similar syntax to C.
- Therefore by knowing how to code in C you would get a head start in learning these other programming languages (e.g. C++, C#, Javascript and PHP)

WHY LEARN C?

- It is also important to know that C can be used in programming most microprocessor based systems.
- The world of robotics, artificial intelligence and embedded systems thrive heavily on efficient programs written in C.
- The next slide shows some examples of these systems.

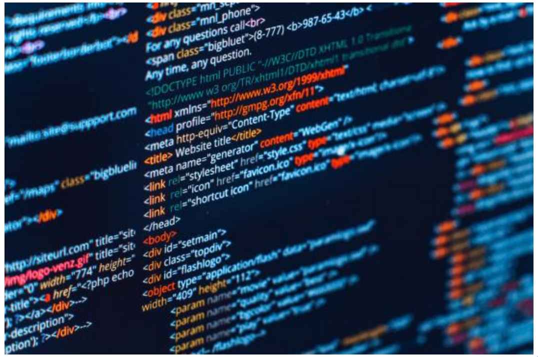
DIFFERENCES



NOKE - SMART LOCK

Noke is a short way of No-Key. With the help of this smart product, you will never have so many keys to open doors.

Go to Product



```
<div class="mnl_phone">
<div class="mnl_phone">
For any questions call<br>
<span class="bigbluet">(8-777) <b>987-65-43</b> <
Any time, any question.
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional
"http://www.w3.org/TR/xhtml1/DTD/xhtml1 transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
<head profile="http://gmpg.org/xfn/11">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title> Website title</title>
<meta name="generator" content="WebGen" />
<link rel="stylesheet" href="style.css" type="text/css" media="screen" />
<link rel="icon" href="favicon.ico" type="image/x-icon" />
<link rel="shortcut icon" href="favicon.ico" type="image/x-icon" />
</head>
<body>
<div id="setmain">
<div class="topdiv">
<div id="flashlogo">
<object type="application/flash" data="paramlogo.swf"
width="409" height="112">
    <param name="movie" value="paramlogo.swf" />
    <param name="quality" value="best" />
    <param name="bgcolor" value="#000000" />
    <param name="play" value="true" />
```

THANKS!

Any questions?

You can find me at

elielkeelson@gmail.com & ekeelson@knust.edu.gh