WHY PROGRAM?

• **Computer** – programmable machine designed to follow instructions

• **Program** – instructions in computer memory to make it do something

• **Programmer** – person who writes instructions (programs) to make computer perform a task

• SO, without programmers, no programs; without programs, a computer cannot do anything
COMPUTER SYSTEMS: HARDWARE AND SOFTWARE

MAIN HARDWARE COMPONENT CATEGORIES

- Central Processing Unit (CPU)
- Main Memory
- Secondary Memory / Storage
- Input Devices
- Output Devices

COMPUTER SYSTEMS: HARDWARE AND SOFTWARE

CENTRAL PROCESSING UNIT (CPU)

Comprised of:

- **Control Unit**
  - Retrieves and decodes program instructions
  - Coordinates activities of all other parts of computer

- **Arithmetic & Logic Unit**
  - Hardware optimized for high-speed numeric calculation
  - Hardware designed for true/false, yes/no decisions
COMPUTER SYSTEMS: HARDWARE AND SOFTWARE

MAIN MEMORY

• It is volatile. Main memory is erased when program terminates or computer is turned off
• Also called Random Access Memory (RAM)
• Organized as follows:
  • bit: smallest piece of memory. Has values 0 (off, false) or 1 (on, true)
  • byte: 8 consecutive bits. Bytes have addresses.
• Addresses
  • Each byte in memory is identified by a unique number known as an address.

• In Figure, the number 149 is stored in the byte with the address 16, and the number 72 is stored at address 23.

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
</tr>
<tr>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
</tr>
</tbody>
</table>

COMPUTER SYSTEMS: HARDWARE AND SOFTWARE

SECONDARY STORAGE

• Non-volatile: data retained when program is not running or computer is turned off
• Comes in a variety of media:
  • magnetic: floppy disk, hard drive
  • optical: CD-ROM, DVD
  • Flash drives, connected to the USB port
COMPUTER SYSTEMS: HARDWARE AND SOFTWARE

Input Devices

- Devices that send information to the computer from outside
- Many devices can provide input:
  - Keyboard, mouse, scanner, digital camera, microphone
  - Disk drives, CD drives, and DVD drives

Software - Programs that run on a computer

- Categories of software:
  - System software:
    - programs that manage the computer hardware and the programs that run on them.
    - Examples: operating systems, utility programs, software development tools
  - Application software:
    - programs that provide services to the user. Examples: word processing, games, programs to solve specific problems
PROGRAMS AND PROGRAMMING LANGUAGES

DEFINITION

- A **program** is a set of instructions that the computer follows to perform a task
- We start with an **algorithm**, which is a set of well-defined steps.

EXAMPLE ALGORITHM FOR CALCULATING GROSS PAY

1. Display a message on the screen asking “How many hours did you work?”
2. Wait for the user to enter the number of hours worked. Once the user enters a number, store it in memory.
3. Display a message on the screen asking “How much do you get paid per hour?”
4. Wait for the user to enter an hourly pay rate. Once the user enters a number, store it in memory.
5. Multiply the number of hours by the amount paid per hour, and store the result in memory.
6. Display a message on the screen that tells the amount of money earned. The message must include the result of the calculation performed in Step 5.
PROGRAMS AND PROGRAMMING LANGUAGES

MACHINE LANGUAGE

• Although the previous algorithm defines the steps for calculating the gross pay, it is not ready to be executed on the computer.

• The computer only executes machine language instructions.

• Machine language instructions are binary numbers, such as

1011010000000101

• Rather than writing programs in machine language, programmers use programming languages.

PROGRAMS AND PROGRAMMING LANGUAGES

TYPE OF LANGUAGES

• Low-level:
  • used for communication with computer hardware directly. Often written in binary machine code (0's/1's) directly.

• High-level:
  • closer to human language
**PROGRAMS AND PROGRAMMING LANGUAGES**

**SOME WELL-KNOWN PROGRAMMING LANGUAGES**

- BASIC
- FORTRAN
- COBOL
- C
- C++
- Ruby
- Java
- C#
- Visual Basic
- JavaScript
- Python
- Visual Basic

**FROM A HIGH-LEVEL PROGRAM TO AN EXECUTABLE FILE**

1. Create file containing the program with a text editor.

2. Run **preprocessor** to convert source file directives to source code program statements.

3. Run **compiler** to convert source program into machine instructions.

4. Run **linker** to connect hardware-specific code to machine instructions, producing an executable file.
   - Steps b–d are often performed by a single command or button click.
   - Errors detected at any step will prevent execution of following steps.
PROGRAMS AND PROGRAMMING LANGUAGES
FROM A HIGH-LEVEL PROGRAM TO AN EXECUTABLE FILE

An integrated development environment, or IDE, combine all the tools needed to write, compile, and debug a program into a single software application.

Examples are Microsoft Visual C++, Turbo C++ Explorer, CodeWarrior, etc.
WHAT IS A PROGRAM MADE OF?
COMMON ELEMENTS IN PROGRAMMING LANGUAGES

- Key Words
- Programmer-Defined Identifiers
- Operators
- Punctuation
- Syntax

```cpp
// This program calculates the user's pay.
#include <iostream>
using namespace std;

int main()
{
    double hours, rate, pay;

    // Get the number of hours worked.
    cout << "How many hours did you work? ";
    cin >> hours;

    // Get the hourly pay rate.
    cout << "How much do you get paid per hour? ";
    cin >> rate;

    // Calculate the pay.
    pay = hours * rate;

    // Display the pay.
    cout << "You have earned $" << pay << endl;
    return 0;
}
```

WHAT IS A PROGRAM MADE OF?
KEYWORDS

- Also known as reserved words
- Have a special meaning in C++
- Can not be used for any other purpose
- Key words in the Program:
  - Using
  - Namespace
  - Int
  - Double
  - return
**WHAT IS A PROGRAM MADE OF?**

**PROGRAMMER-DEFINED IDENTIFIERS**

- Names made up by the programmer
- Not part of the C++ language
- Used to represent various things:
  - variables (memory locations)
  - functions, etc.
- In Program:
  - Hours
  - Rate
  - pay

```cpp
// This program calculates the user's pay.
#include <iostream>
using namespace std;

int main()
{
    double hours, rate, pay;
    // Get the number of hours worked.
    cout << "How many hours did you work? ";
    cin >> hours;
    // Get the hourly pay rate.
    cout << "How much do you get paid per hour? ";
    cin >> rate;
    // Calculate the pay.
    pay = hours * rate;
    // Display the pay.
    cout << "You have earned $" << pay << endl;
    return 0;
}
```

**OPERATORS**

- Used to perform operations on data
- Many types of operators:
  - Arithmetic - ex: +, -, *, /
  - Assignment - ex: =
- Some operators in Program:
  - <<
  - >>
  - =
  - *
WHAT IS A PROGRAM MADE OF?

PUNCTUATION

• Characters that mark the end of a statement, or that separate items in a list
• In Program:
  • *
  • ,
  • ;

```
1 // This program calculates the user's pay.
2 #include <iostream>
3 using namespace std;
4
5 int main()
6 {
7    double hours, rate, pay;
8    // Get the number of hours worked.
9    cin >> hours;
10   // Get the hourly pay rate.
11   cout << "How much do you get paid per hour? ";
12   cin >> rate;
13   // Calculate the pay.
14   pay = hours * rate;
15   // Display the pay.
16   cout << "You have earned $" << pay << endl;
17   return 0;
20 }
```

WHAT IS A PROGRAM MADE OF?

SYNTAX

• The rules of grammar that must be followed when writing a program
• Controls the use of key words, operators, programmer-defined symbols, and punctuation
WHAT IS A PROGRAM MADE OF?

VARIABLES

• A variable is a named storage location in the computer’s memory for holding a piece of data.

• In Program we used three variables:
  • The `hours` variable was used to hold the hours worked
  • The `rate` variable was used to hold the pay rate
  • The `pay` variable was used to hold the gross pay

WHAT IS A PROGRAM MADE OF?

VARIABLE DEFINITIONS

• To create a variable in a program you must write a variable definition (also called a variable declaration)

• Here is the statement from Program that defines the variables:
  • `double hours, rate, pay;`

• There are many different types of data, which you will learn about in this course.
• A variable holds a specific type of data.
• The variable definition specifies the type of data a variable can hold, and the variable name.

• Once again, line 7 from Program:
  • `double hours, rate, pay;`
• The word `double` specifies that the variables can hold double-precision floating point numbers. (You will learn more about that in next lecture)
INPUT, PROCESSING, AND OUTPUT

THREE STEPS THAT A PROGRAM TYPICALLY PERFORMS:

1. Gather input data:
   • from keyboard
   • from files on disk drives

2. Process the input data

3. Display the results as output:
   • send it to the screen
   • write to a file

THE PROGRAMMING PROCESS

1. Clearly define what the program is to do.
2. Visualize the program running on the computer.
3. Use design tools such as a hierarchy chart, flowcharts, or pseudocode to create a model of the program.
4. Check the model for logical errors.
5. Type the code, save it, and compile it.
6. Correct any errors found during compilation. Repeat Steps 5 and 6 as many times as necessary.
7. Run the program with test data for input.
8. Correct any errors found while running the program.
    Repeat Steps 5 through 8 as many times as necessary.
9. Validate the results of the program.
PROCEDURAL AND OBJECT-ORIENTED PROGRAMMING

- **Procedural programming:**
  - focus is on the process.
  - Procedures/functions are written to process data.

- **Object-Oriented programming:**
  - focus is on objects,
  - which contain data and the means to manipulate the data.
  - Messages sent to objects to perform operations.