Computer Software

The first computer programs ever written were for a mechanical computer designed by Charles Babbage in the mid-1800s. The person who wrote these programs was Ada Lovelace, who was a talented mathematician. Thus, she is referred to as “the first computer programmer.” This section discusses fundamental issues of computer software.

Computer software is a set of program instructions, including related data and documentation, that can be executed by computer. This can be in the form of instructions on paper, or in digital form. While system software is intrinsic to a computer system, **application software** fulfills users’ needs, such as a photo-editing program.

Part I: Syntax, Semantics, and Program Translation

Programming languages (called “artificial languages”) are languages just as “natural languages” such as English and Mandarin (Chinese). Syntax and semantics are important concepts that apply to all languages.

The **syntax** of a language is a set of characters and the acceptable arrangements (sequences) of those characters. English, for example, includes the letters of the alphabet, punctuation, and properly spelled words and properly punctuated sentences. The following is a syntactically correct sentence in English: “Hello there, how are you?”

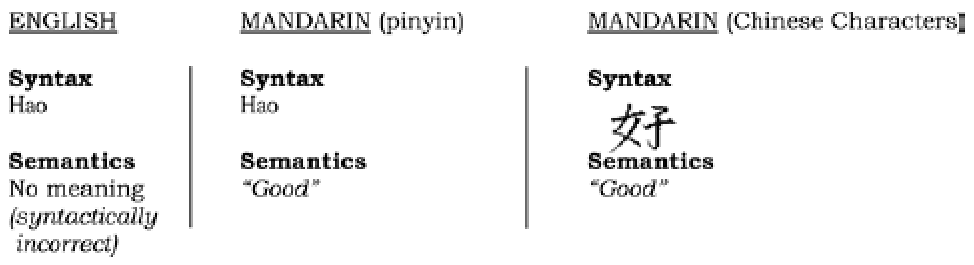
The following, however, is not syntactically correct: “Hello there, hao are you?”

In this sentence, the sequence of letters “hao” is not a word in the English language. Now consider the following sentence,

“Colorless green ideas sleep furiously.”

This sentence is syntactically correct, but is **semantically** incorrect, and thus has no meaning.

The **semantics** of a language is the meaning associated with each syntactically correct sequence of characters. In Mandarin, “Hao” is syntactically correct meaning “good.” (“Hao” is from a system called pinyin, which uses the Roman alphabet rather than Chinese characters for writing Mandarin.) Thus, every language has its own syntax and semantics, as shown below.

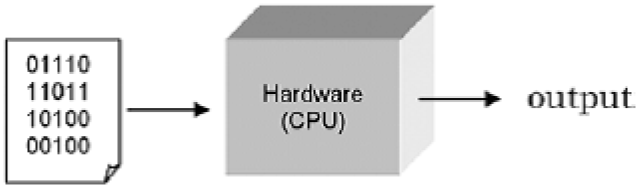


ENGLISH	MANDARIN (pinyin)	MANDARIN (Chinese Characters)
Syntax Hao	**Syntax** Hao	**Syntax** 好
Semantics No meaning *(syntactically incorrect)*	**Semantics** *“Good”*	**Semantics** *“Good”*

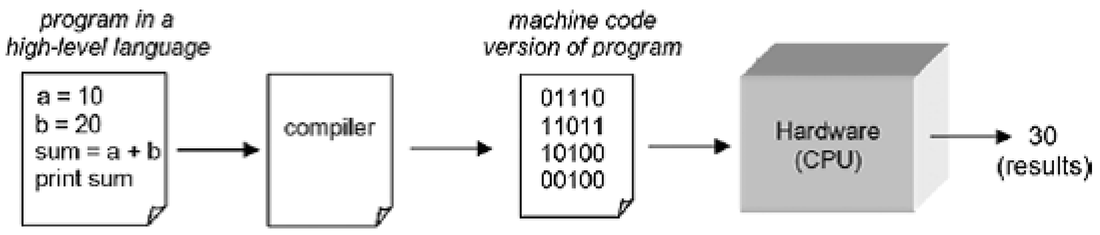
1. Write your own syntactically correct sentence, that has no meaning.

Part II: Program Translation

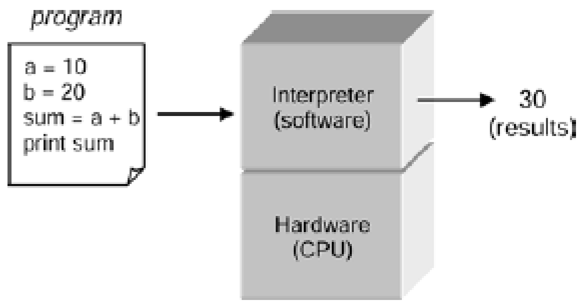
A central processing unit (CPU) is designed to interpret and execute a specific set of instructions represented in binary form (i.e., 1s and 0s) called machine code. Only programs in **machine code** can be executed by a CPU, as shown below.



Writing programs at this “low level” is tedious and error-prone. Therefore, most programs are written in a “high-level” programming language such as Python. Since the instructions of such programs are not in machine code that a CPU can execute, a translator program must be used. There are two fundamental types of translators. One, called a **compiler**, translates programs directly into machine code to be executed by the CPU, as shown below.



The other type of translator is called an **interpreter**, which executes program instructions in place of (“running on top of”) the CPU, as shown below.



```
a = 10
b = 20
sum = a + b
print sum
```

Thus, an interpreter can immediately execute instructions as they are entered. This is referred to as **interactive mode**. This is a very useful feature for program development. Python, as we shall see, is executed by an interpreter. On the other hand, compiled programs generally execute faster than interpreted programs. Any program can be executed by either a compiler or an interpreter, as long there exists the corresponding translator program for the programming language that it is written in.

Part III: Program Debugging: Syntax Errors vs. Semantic Errors

Program debugging is the process of finding and correcting errors ("bugs") in a computer program. Programming errors are inevitable during program development. **Syntax errors** are caused by invalid syntax (for example, entering prnt instead of print). Since a translator cannot understand instructions containing syntax errors, translators terminate when encountering such errors indicating where in the program the problem occurred.

In contrast, **semantic errors** (generally called logic errors) are errors in program logic. Such errors cannot be automatically detected, since translators cannot understand the intent of a given computation. For example, if a program computed the average of three numbers as follows, (num1 + 1 num2 + 1 num3) / 2.0

a translator would have no means of determining that the divisor should be 3 and not 2. Computers do not understand what a program is meant to do, they only follow the instructions given. It is up to the programmer to detect such errors. Program debugging is not a trivial task, and constitutes much of the time of program development.

Programming languages fall into a number of programming paradigms. The two major programming paradigms in use today are procedural (imperative) programming and object-oriented programming. Each provides a different way of thinking about computation. While most programming languages only support one paradigm, Python supports both procedural and object-oriented programming. We will start with the procedural aspects of Python.

Concepts and Procedures

1. Two general types of software are system software and _______________ software.

2. The syntax of a given language is,
    a. the set of symbols in the language.
    b. the acceptable arrangement of symbols.
    c. both of the above

3. The semantics of a given language is the meaning associated with any arrangement of symbols in the language. (TRUE/FALSE)

4. CPUs can only execute instructions that are in binary form called _____________.
5. The two fundamental types of translation programs for the execution of computer programs are _________________ and ___________________.

6. The process of finding and correcting errors in a computer program is called __________.
7. Which kinds of errors can a translator program detect?
    a. Syntax errors
    b. Semantic errors
    c. Neither of the above
8. Two major programming paradigms in use today are ______________ programming and _____________________ programming.

Problem Solving

1. Give two specific examples of an application program besides those mentioned in this activity.

2. For each of the following statements in English, indicate whether the statement contains a syntax error, a logic (semantic) error, or is a valid statement.
    a. Witch way did he go?
    b. I think he went over their.
    c. I didn’t see him go nowhere.

3. For each of the following arithmetic expressions for *adding up the integers 1 to 5*, indicate whether the expression contains a syntax error, a semantic error, or is a valid expression.
    a. 1 + 2 ++ 3 + 4 + 5
    b. 1 + 2 + 4 + 5
    c. 1 +2 + 3 + 4 + 5
    d. 5 + 4 + 3 + 2 + 1

4. Give one benefit of the use of a compiler, and one benefit of the use of an interpreter.