**Reading Comprehension**

1)- Which is the goal of the book?

The goal of the book is to teach to think like a computer scientist.

2)- Which is the most important skill of a computer scientist?

The most important skill of a computer scientist is problem solving. Problem solving means the ability to formulate problems, think creatively about solutions, and express a solution clearly and accurately.

3)- Which are the differences between **high-level languages** and **low-level languages**? Define them and give examples.

computers can only execute programs written in

low-level languages. Thus, programs written in a high-level language have to be

processed before they can run. This extra processing takes some time, which is

a small disadvantage of high-level languages.

But the advantages are enormous. First, it is much easier to program in a

high-level language. Programs written in a high-level language take less time

to write, they are shorter and easier to read, and they are more likely to be

correct. Second, high-level languages are portable, meaning that they can

run on different kinds of computers with few or no modifications. Low-level

programs can run on only one kind of computer and have to be rewritten to run

on another.

Due to these advantages, almost all programs are written in high-level languages.

Low-level languages are used only for a few specialized applications.

4)- Describe the functions of **interpreters** and **compilers**.

interpret: To execute a program in a high-level language by translating it one

line at a time.

compile: To translate a program written in a high-level language into a low-

level language all at once, in preparation for later execution.

5)- What is a **program** and what is a **computation**?

program: A set of instructions that specifies a computation.

The computation might be something mathematical, such as solving

a system of equations or finding the roots of a polynomial, but it can also be a

symbolic computation, such as searching and replacing text in a document or

(strangely enough) compiling a program.

6)- What are **bugs** and what is **debugging**? Why are **programming** and **debugging** thought to be the same thing?

programming errors are called bugs and the process of tracking them down and correcting them is called debugging.

For some people, programming and debugging are the same thing. That is, programming is the process of gradually debugging a program until it does.

7)- Syntax

1. Give an example of syntax error in English.

Page 4, line 24. “.this sentence…”.

1. Give an example of syntax error in Python.

Page 8, line 9. “Hello, World!”. Should (“Hello World!”).

1. Why is a syntax error "more important" in programming than in English?

Page 4, line 28. Because a few syntax error in English are not a significant problem, you can read the text, “If there is a single syntax error anywhere in your program, Python will print an error message.”

8)- What is a **runtime error**?

An error that does not occur until the program has started to execute but that prevents the program from continuing.

9)- What is a **semantic error**?

An error in a program that makes it do something other than what the programmer intended.

11)- Why is English a **natural language**? What type of language is Python?

Natural languages are the languages that people speak. Because they evolved naturally.

Python is a Programming languages are formal languages that have been designed to express computations.

12)- Which are the two basic **syntax rules**? Explain them.

Syntax rules come in two flavors, pertaining to tokens and structure.

One of the basic elements of the syntactic structure of a program, analogous to a word in a natural language.

Tokens are the basic elements of the language, such as words, numbers, and chemical

elements.

the structure of a statement—that is, the way the tokens are arranged.

13)- What is **parsing**? Why do humans (as speakers) and computers need to do that?

When you read a sentence in English or a statement in a formal language, you

have to figure out what the structure of the sentence is (although in a natural

language you do this subconsciously). This process is called parsing. No ES ESTo

The process when you read a sentence in English or a statement in a formal language is called parsing. Humans and computers have to figure out what the structure of the sentence is (although in a natural language you do this subconsciously).

14)- Complete the following chart with the main differences between **natural languages** and **formal languages**

|  |  |  |
| --- | --- | --- |
|  | **Natural Languages** | **Formal languages** |
| **Ambiguity** | Yes | No |
| **Redundancy** | Yes | No |
| **Literalness** | No | Yes |

15)- According to the authors, which is the best way of reading a program in distinction to reading in English?

Here are some suggestions for reading programs (and other formal languages).

First, remember that formal languages are much more dense than natural languages, so it takes longer to read them. Also, the structure is very important, so 8 The way of the program it is usually not a good idea to read from top to bottom, left to right. Instead,

learn to parse the program in your head, identifying the tokens and interpreting

the structure. Finally, the details matter. Little things like spelling errors and

bad punctuation, which you can get away with in natural languages, can make

a big difference in a formal language.

**Grammar. Remember to quote the examples and to indicate page and line number.**

1)- Taking the first part of the text (“*The way of the program*”):

1. Which is its main verb tense? Are there any other verb tenses?

The main verb tense is Simple Present. Yes, there are future (Page 1, line 14) and passive voice (Page 1, line 12).

1. Identify the main ideas.

The main ideas are to teach you to think like a computer scientist, to learn the process of programming and to practices problem solving skills.

1. Identify examples and less relevant information.

Page 1, line 3. “Like mathematics,…”, “Like scientist,…” and “Like engineers,…”

1. Find examples of the different parts of speech: verbs, nouns, adjectives and adverbs. (3 examples from each)

Verbs: Page 1, line 1: Is (be) and think. Page 1, line 4: Design.

Nouns: Page 1, line 1: Goal, book and scientist.

Adjectives: Page 1, line 2: Best. Page 1, line 3: Formal. Page 1, line 3: Important.

Adverbs: Page 1, line 4: Specifically. Page 1, line 9: Creatively. Page 1, line 10: Crearly.

2)- Identify all the connectors under the headings discussed in class. (Listing and sequencing; giving examples; giving reason and cause; indicating addition; making contrasts, explaining results or effects). Give at least two examples of each type.

Indicating addition: Page 1, line 3: “Like mathematics,…”, “Like scientist,…”.

Indicating result: Page 1, line 10: As it turns. Page 1, line 14: As we go along.

Reason: Page 1, line 12: That´s why.

Sequencing: Page 1, line 13: On one level. On another level.

3)- List all the types of modal verbs you can find. What are they expressing? (Ability, prohibition, lack of necessity, etc. Provide at least 5 different types)

Can, Ability: Page 2, line 1.” Loosely speaking, computers can only execute programs written in low-level languages.”

Prohibition:

Lack of necessity:

Must: Page 2, line 2. “Have to be…” Page 4, line 24. “…,must begin…”

Should: Migth, Probability: Page 1, line 17. “..;other high-level languages you might have heard of are C, C++, Perl, and Java.” Page 3, line 24 “…,you sould store it…”

4)- Find examples of sentences in passive voice. In which verb tense are they written? (At least two examples)

(Page 1, line 12). Is Called. Simple Present.

(Page 1, line 12). “…,almost all program are written…”

5)- Find examples of conditional sentences. Which type are they? (At least two examples)

Page 5, line 7. “If there is a semantic error in your program, it will run successfully, in the sense that the computer will not generate any error messages, but it will not do the right thing.” Type 1.

Page 5, line 21. “If your hypothesis was correct, then you can predict the result of the modification, and you take a step closer to a working program.” Type 2.

6)- Find examples of relative clauses. (At least two examples)

Page 1, line 12. “That´s why this chapter is called…”

Page 3, line 26. “A program is a sequence of instructions that specifies how to perform…”

7)- Find examples of present perfect tense. Are they in simple or continuous (progressive) aspect? (At least two examples).

Page 4, line 7. “Every program you´ve ever used,…” Present Perfect Simple.

Page 5, line 3. “…that something exceptional (and bad) has happened.” Present Perfect Simple.