**Cuestionario Java**

1. **¿ En que lenguaje pueden las computadoras corer los programas?**

Machine code o de ensamblaje

1. **¿Cuáles son las dos principales ventajas de los niveles de alto nivel?**

Son mucho mas faciles de programar y tienen portabilidad.

1. **¿ Cual es la ventaja de Java sobre otros lenguajes de alto nivel?**

Compila a bytecode el source code, el cual es facil de interpretar en cualquier computadora (solo tiene que tener el interprete de Java).

1. **¿Cuáles son las dos formas de traducir el lenguaje de alto nivel a de machine code?**

Interpretar y Compilar

1. **¿Java es interpretado o compilado?**

Ambos

1. **¿Qué es un programa?**

Un programa es una secuencia de instrucciones que especifica cómo realizar un cálculo.

1. **¿Cómo se le dice a las instrucciones?**

Statements

1. **¿Cuáles son los statements basicos que todo lenguaje de alto nivel es capaz de realizar?**

Input, Output, Math, Testing(probar si se cumple una condicion) y Repetition

1. **¿Qué es debugging?**

Encontrar y solucionar errores (bugs)

1. **¿Cuáles son los tipos de errores que pueden surgir en un programa?**
2. Syntax (Estructura del programa y sus reglas).
3. Run-time errors (conocidos como exceptcions en Java, ocurren hasta que corres el programa).
4. Logic errors and semantics (El programa no realiza lo que querias)

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

Syntax rules come in two flavors, pertaining to tokens and structure. Tokens are the basic elements of the language, like words and numbers and chemical elements

The second type of syntax rule pertains to the structure of a statement; that is, the way the tokens are arranged.

**ambiguity**: Natural languages are full of ambiguity, which people deal with by using contextual clues and other information. Formal languages are designed to be unambiguous, which means that any statement has exactly one meaning, regardless of context.

**redundancy**: To make up for ambiguity and reduce misunderstandings, natural languages are often redundant. Formal languages are more concise. literalness: Natural languages are full of idiom and metaphor. Formal languages mean exactly what they say.

**Programs**: The meaning of a computer program is unambiguous and literal, and can be understood entirely by analysis of the tokens and structure.

A library is a collection of class and method definitions.

***1.6 Glossary***

**problem-solving:** The process of formulating a problem, finding a solution, and expressing the solution.

**high-level language:** A programming language like Java that is designed to be easy for humans to read and write.

**low-level language:** A programming language that is designed to be easy for a computer to run. Also called “machine language” or “assembly language.”

**formal language:** Any of the languages people have designed for specific purposes, like representing mathematical ideas or computer programs. All programming languages are formal.

**natural language:** Any of the languages people speak that have evolved naturally.

**portability:** A property of a program that can run on more than one kind of computer.

**interpret:** To run a program in a high-level language by translating it one line at a time.

**compile:** To translate a program in a high-level language into a low-level language, all at once, in preparation for later execution.

**source code:** A program in a high-level language, before being compiled.

**object code:** The output of the compiler, after translating the program.

**executable:** Another name for object code that is ready to run.

**byte code:** A special kind of object code used for Java programs. Byte code is similar to a low-level language, but it is portable, like a high-level language.

**statement:** A part of a program that specifies a computation.

**print statement:** A statement that causes output to be displayed on the screen.

**Comment**: A part of a program that contains information about the program, but that has no effect when the program runs.

**method:** A named collection of statements.

**library**: A collection of class and method definitions.

**bug**: An error in a program.

**syntax**: The structure of a program. semantics: The meaning of a program.

**parse**: To examine a program and analyze the syntactic structure.

**syntax error:** An error in a program that makes it impossible to parse (and therefore impossible to compile).

**exception:** An error in a program that makes it fail at run-time. Also called a run-time error.

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

**debugging:** The process of finding and removing any of the three kinds of errors.