Universidad politécnica del estado de yucatan

Ingeniería en robotica computacional

Group:A

Full name: Jose Javier Pacheco Tun

Date: 05/09/2021

Structured programming

HOMEWORK 1

Stages of program compilation:

Compiler:

It is a translator program whose function is to translate (compile) a source program written in some high-level language into machine language. This translated program or object program is normally stored in secondary memory in executable form and is loaded into main memory each time it needs to be executed.

Interpreter :

Like the compiler, the interpreter translates a source program written in some high-level language, but with the difference that each instruction is executed immediately, without generating a machine language program. Compilation is generally a more efficient process than interpretation. This is because the sentences within a loop must be reinterpreted each time they are executed by an interpreter. In contrast with a compiler, each statement is translated into machine language only once.

PHASES OF A COMPILER

Preprocessing: Transformations to the Source File, prior to Compilation.

Lexical Analysis: Recognition of the Elements of Language.

Syntactic Analysis: Recognition of the Structure of Language.

Semantic Analysis: Recognition of the coherence of the Entry.

Generation of Intermediate Code: Transformation of the Input into a representation of intermediate code for an abstract machine.

Code Optimization: Improvements to intermediate rendering that result in faster code to execute.

Code Generation: Transformation of intermediate code into object code.

Debugging: Error Recognition.

Symbols Table Administration: Recognition of the names of the identifiers used in the Entry and their different attributes

These phases are not strictly sequential, but rather the modules that implement them interact during the compilation process, complementing their tasks with each other. However, as a didactic, a sequential scheme of phases is proposed, which will allow the study of each module independently.

Levels of programming

LOW LEVEL: A low-level or first-generation feature programming language is one in which its instructions exert direct control over the hardware and are conditioned by the physical structure of the computers that support it.

The use of the word low in its name does not imply that the language is less powerful than a high-level language, but rather refers to the reduced abstraction between the language and the hardware.

MID-LEVEL: Middle-level language is a computer programming language like the C language, which are among the high-level languages and the low-level languages.

They are often classified as high-level, but allow certain low-level handling. They are accurate for certain applications such as creating operating systems, as they allow abstract handling (independent of the machine, unlike the assembler), but without losing much of the power and efficiency.

HIGH LEVEL: The high-level language is one that is closer to human natural language than to the binary language of computers, which is known as low-level language.

Its main function is that from its development, there is the possibility that the same program can be used on different machines, that is, it is independent of a specific hardware. The only condition is that the PC has a program known as

translator or compiler, which translates it into language.

Info 1:

https://kataix.umag.cl/~jaguila/Compilers/T01_Fases_Compilador.pdf

https://nivelesdeprogramacion.blogspot.com/2019/07/niveles-de-programacion.html#:~:text=1.,del%20microprocesador%20de%20un%20ordenador