|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 🔑 **Essential** 🔑 | | | |  | 🏆 **Enhanced** 🏆 | | | |
|  |  |  |  |  |  |  |  |  |
| \_\_\_\_\_ | \_\_\_\_\_ | \_\_\_\_\_ | \_\_\_\_\_ |  | \_\_\_\_\_ | \_\_\_\_\_ | \_\_\_\_\_ | \_\_\_\_\_ |

**Score: \_\_\_\_\_**

**14 – Translation and Interpretation**

**Activities**

COMP256 – Computing Abstractions

Dickinson College

Spring 2022

Prof. Grant Braught

**Name:**

Today’s class introduced the new topic of Language Abstractions. Each level in the language abstraction hierarchy defines a new abstract machine that can be programmed at its level of abstraction. We first saw this with machine language in the Machine Abstractions topic. Programming at the machine language level is an abstraction that allows us to focus more on the details that matter to us than when we were programming at the microinstruction level. For example, at the machine language level we can think about what opcode and operands we need while ignoring the microinstruction level details about things like which switches need to be opened or closed. We let the abstraction handle that for us!

We will now begin looking at machines at higher levels of abstraction (Assembly Language and High-level Language) and how programs written for these machines can be executed on a physical machine. Translation (compilation/assembly) and interpretation are the two primary mechanisms by which this happens. Though we will also see that a combination of the two is increasingly popular and has some of the advantages of each. Through the labs you will gain hands-on experience with how interpretation and translation work by writing programs that perform these tasks.

**Translation and Interpretation:**

Each of the following videos provides some additional information and details about the processes, advantages and disadvantages of program translation and program interpretation. They are relatively short, and it is worth watching all three. The different perspectives and takes will help to develop and reinforce the main ideas. As you watch these identify the similar concepts across them and pay attention to the slightly different terminology that gets used by different people.

* The video *Compiler and Interpreter: Compiled Language vs Interpreted Programming Languages* from Coding Mentors is a good introduction to the topic.
  + <https://www.youtube.com/watch?v=I1f45REi3k4> (6:04)
* The video *Compilation vs Interpretation* from BGZDevTips covers the same topics but with a slightly different take and some more illustrations.
  + <https://www.youtube.com/watch?v=JNMy969SjyU> (6:06)
* Jakob, on his blog Finematics provides some reading and a video *Compiled vs Interpreted Programming Languages*. This is probably the most comprehensive discussion and he writes everything out if you find that easier to follow. He also does a more complete job of explaining the virtual machine approach and Just In Time (JIT) compilation.
  + <https://finematics.com/compiled-vs-interpreted-programming-languages/> (9:33)

🔑 1. There are a number of key terms that are used throughout these videos. Draw on the content of these videos and from class to give a brief definition of each of the following terms using a few sentences of your own words.

a. Source Code

b. Executable (or binary) File

c. Intermediate Language (or Representation)

🔑 2. There are a number of different programs involved in translation and interpretation. Draw on the content of the videos and from class to describe what each of the following programs does using a few sentences of your own words.

a. Compiler

b. Assembler

c. Interpreter

d. Virtual Machine

🔑 3. The two primary approaches to the execution of high-level language programs, translation and interpretation, have their own advantages and disadvantages. Several of the videos above discuss these advantages and disadvantages. Draw on the information in those videos to place each advantages or disadvantage below into the appropriate cell in the table (e.g. place “Slower Program Execution” in one cell and “Faster Program Execution” in another cell.

* Slower Program Execution / Faster Program Execution
* Slower Development & Debugging / Faster Development & Debugging
* Source code is private / Source code is public
* Programs are Cross Platform (portable) / Programs are Platform Dependent (not portable)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  | **Advantages** | **Disadvantages** |  |
|  | **Translated** |  |  |  |
|  | **Interpreted** |  |  |  |
|  |  |  |  |  |

🔑 4. For each of the following situations, weigh the above advantages and disadvantages and indicate if a translated language or an interpreted language would be more appropriate. **Give a brief rationale for your choice**.

a. A small utility program that will be used by people with lots of different types of computers.

b. A program that the end users should be able to easily change to suit their specific situation.

c. A program that is based on trade secrets that you want people to run but not modify.

d. A program to perform complex calculations on a high-performance super-computer.

e. A program that needs to be changed, updated and tested frequently.

**The Hybrid Approach:**

The intermediate or hybrid approach was pioneered by Java and the engineers at Sun Microsystems to try to capture some of the advantages of both translation and interpretation, while mitigating their disadvantages. It has since been applied to a wide variety of popular languages including C#, Python, Ruby, Kotlin and many others.

🔑 5. The hybrid approach combines both translation and interpretation with the translation taking place on the software developer’s machine and interpretation taking place on the end user’s machine.

a. Write a sentence using the terms **source code, compiler and intermediate language** that describes what happens on the developer’s machine during the translation stage in the hybrid approach.

b. Write a sentence using the terms **intermediate language (or bytecode), virtual machine, interpreter and machine language** that describes what happens on the end user’s machine during the interpretation stage in the hybrid approach**.**

🏆 6. In a few sentences of your own words, explain how the hybrid approach captures some of the benefits of both interpreted programs and translated programs.

🔑 7. For each of the following, indicate if it is most likely to be source code, bytecode or executable code.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  | Type of Code |  |
|  | a. A program written in the interpreted language BASIC. |  |  |
|  | b. The program that is the interpreter for BASIC programs. |  |  |
|  | c. A program written in the Java language before it is compiled. |  |  |
|  | d. A compiled Java program. |  |  |
|  | e. The program that is the Java Compiler. |  |  |
|  | f. The program that is the Java Virtual Machine. |  |  |
|  | g. A program written in the C++ language before it is compiled. |  |  |
|  | h. The program that is the C++ Compiler. |  |  |
|  | i. A compiled C++ program. |  |  |
|  |  |  |  |

🏆 🏆 8. Recall that some of the performance improvement that is obtained by the use of processor cache was due to the fact that programs tend to exhibit temporal locality. Discuss how the concept of temporal locality is also important to the performance improvements obtained by virtual machines that use Just in Time (JIT) compilation. Note: JIT compilation is discussed in the final of the 3 videos.

**WiD Assignment and Repository:**

COMP 256 is part of the Writing in the Discipline (WiD) thread for the computer science major. Each of the core courses in the major contains a WiD learning goal. By completing all of the WiD assignments across all of the core courses and collecting that work into a repository you will satisfy the College’s WiD graduation requirement.

You will begin the WiD assignment for COMP 256 next week. For this course the WiD learning goal is:

* “Be able to use metaphor and analogy to explain complex technical concepts.”

This seems a suitable goal for this course as metaphors and analogies are forms of abstraction themselves.

🔑 9. To prepare for that assignment, ensure that you are able to access your WiD repository on GitHub. Log into GitHub, find your WiD repository and give its URL here.

Optional: To help me improve and scope these activities for future semesters please consider providing the following feedback.

a. Approximately how much time did you spend on this activity outside of class time?

b. Please comment on any particular challenges you faced in completing this activity.