**CS365 Final**

**Spring 2020**

**I state that this is my work, and only my work, and that I did not receive any assistance, in any form, from anyone else, student or otherwise. I understand that failure to comply with this statement may result in an ‘F’ for the class and that I may be reported to the Dean of Students for academic dishonestly.**

**Name Elena Corpus Date** 5/13/2020

**10 points each. You need to answer 10 of the following questions. You must place an X in the response area for the question you choose not to answer.**

1. **What are the 2 most common means of passing arguments to parameters discussed in class? List strength and weakness of each of both approaches.**

In mode: argument is passed into the parameter

Strength: fast process

Weakness: one is required for each parameter thus making it larger

Out Mode: the value of the parameter is passed back to the argument

Strength: fast process

Weakness: there is a potential for a parameter collision if the same variable is passed twice in the same program

1. **Where the previous question asked about *how* an argument is passed to a parameter, in this question list several approaches different languages use to create “flexibility” regarding *what*, or *how many* arguments are passed to the method. Your answer will include approaches from more than one language.**

C: only the pass by value, it can achieve the pass-by-reference when going to an argument. Thus, allows for typing parameters as pointers to constants.

C++: it only allows for the pointer type as a reference type; thus they are dereferenced in the function.

Java: only passed by value, and when the references to objects are passed, they are passed by value also.

C#: allows for both pass-by-value and pass-by-reference, but it also supports out-mode parameters. Parameter types in the method header need to be preceded by the reserved word ref as well.

1. **What are the goals of object-oriented programming? Describe each goal, and how it (supposedly) improves the software development process.**

Definition: define the data type of a data structure and the types of operations that can be applied to the structure.

The goals of object-oriented programming include:

Reusability:

Define: this is for the ability to use this for other programs as well, make the basic structure and we would be able to use it for other things and add to it as well to make it user specific.

Improves:

Extensibility:

Define: to be able to extend the life of … or stretched out

Improves: this improves the software development process because if ensure that programs have the ability be stretched out for long periods of time we will not have to continue to keep watch over it and keep error checking if it is going to fail because of new updates.

Flexibility:

Define: able to make changes and modifications to the program without having to modify everything, kind of like using global and local variables

Improves: the adaptability of the programs because some programs will have the ability to be used in different places or different things.

1. **You cannot create objects from an interface. What is an interface, and why use an interface if you cannot create an object from it?**

Interface: it is a connection between classes

We would use an interface even though we cannot create objects from it because they are not strong. Interfaces only contain abstract methods, and those methods do not have a body. But we would still use them because interfaces can create multiple classes, so then we are able to edit those and make them into what we need.

1. What is a “referentially transparent function?” What is programming in a "purely functional style"? How is programming in a purely function style achieved?

X

1. **Most languages overload at least one operator. Some languages, such as C++, allow for user-defined overloaded operators, while other languages, such Java, do not allow this to happen. What are the advantages and disadvantages of allowing operators to be overloaded?**

Advantages:

The readability is great, like when doing “x + y” rather than having to express it as x.add(y), the first example is more readable than the second.

We can create more than one function with the same name

Disadvantages:

Even with the smallest mix up causes a larger mistake, like when putting two strings together using “+,” you lose the space in between the strings like it does in other languages. Thus, making developers believe that there is not guarantee all will match the functionality that we expect them too.

1. **An ArrayList provides a convenient means of creating an "array like" data structure that can grow and shrink, yet provide "array like" direct access to the data through set and get methods. What we know about how arrays work tells us that the array cannot just grow or shrink as needed. Why would the ArrayList still need to have a basic array structure and why can’t it really grow or shrink?**

ArrayList : resizable array (dynamically)

It still needs the basic array structure because although when working in java you are calling to import you are still using an array to collect all that data because that is the only structure that allows for this is allowed and needed.

1. What type of information goes into an activation record or a stack frame? When does the system create an activation record? When does the system release, or free, and activation record?

Contents of an Activation record includes, return values, parameters, saved machine status, local data.

The system creates an activation record when the procedure is called.

The activation record is released when the function is completed, executed and returns the value.

1. **What is the difference between overloading and overriding, and when would you use each?**

Overload: Assigning a new meaning to something

Overloading would be used when you have two functions under the same name, however each have different parameters. Overloading would be used when there is a new set of parameters added to the function.

Override: replacing the meaning to something

Overriding is when there are two or more functions that have the same name and parameters; however, they are in two different sections of the program. Thus, meaning that the data will get mess up depending on the steps the program takes to run and what function it goes thru first and last. It will be used when declaring another function in another method/class in addition to the one in the main class.

1. **Define “strongly-typed” and “weakly-typed” languages. Give an example of a strongly typed language. Describe several advantages of a strongly typed language.**

Strongly typed languages are the ones that variables need specific data types, without such, it will result in errors, type errors. There must be some identification of data types in order to ensure that expressions match up. However, the compiler is set in place to avoid this confusion and blocks the program from running because it notes what data type each variable is.

Weakly typed languages are ones that are not bounded to a specific data types, they however still have types, but there are not as many safety measures out to protect the variables in comparison to the strongly types languages.

Examples of Strongly Type: Python, Java

Advantages of Strongly Type:

Error detection is much earlier, thus provides quicker speeds in development.

Because each data type is predetermined, there is no run-time penalties.

1. **Some languages include a return value (including void) in the method signature, while others do not. What is the method signature? What are the differences in the languages that allow for this distinction? What are the advantages and disadvantages of these approaches?**

Method Signature: the information that is needed to invoke that specific method. (Ie Method Name and arguments that will be passed thru it.)

Differences in Languages: Depending on the language, depends on what is necessary for its method signature, each require the method name and its arguments, however, some languages allow for the return of the values from the method, and some do not.

Advantages:

It can and will return functions and variables that were passed thru the value, or that were declared or processed in the function.   
Disadvantages:

This cannot be used with large structures