Alexander J King

CPL

Homework #10

4.13.20

1)What is abstraction?

An abstraction is a view or representation of an entity that includes only the most significant attributes

2)What is an abstraction data type?

An abstract data typeis a user-defined data type that satisfies the following two conditions:–The representation of objects of the type is hidden from the program units that use these objects, so the only operations possible are those provided in the type's definition–The declarations of the type and the protocols of the operations on objects of the type are contained in a single syntactic unit. Other program units are allowed to create variables of the defined type.

3)What are advantages of data abstraction?

Advantages the first condition –Reliability--by hiding the data representations, user code cannot directly access objects of the type or depend on the representation, allowing the representation to be changed without affecting user code–Reduces the range of code and variables of which the programmer must be aware–Name conflicts are less likely•Advantages of the second condition–Provides a method of program organization–Aids modifiability (everything associated with a data structure is together)–Separate compilation

4)Explain how the stack works?

LIFO- Last in first out

Push – Add item to top

Pop – Remove item from top

5)In C++ what is a Friend class or Friend Function? A friend class can acccess private and or protected members of other classes in which it is declared as a friend. A friend function like the friend class can be given access to private and protected functions. This could be a method of another class or a global function.

6)Explain how each of the following languages handle Information hiding, Constructors and destructors (give examples).

a.C++

•Constructors:–Functions to initialize the data members of instances (they do notcreate the objects)–May also allocate storage if part of the object is heap-dynamic–Can include parameters to provide parameterization of the objects–Implicitly called when an instance is created–Can be explicitly called–Name is the same as the class name

•Destructors–Functions to cleanup after an instance is destroyed; usually just to reclaim heap storage–Implicitly called when the object’s lifetime ends–Can be explicitly called–Name is the class name, preceded by a tilde (~)

User()

{

std::cout << “Constructor\n”

}

User (std:string first\_name, std::string last\_name)

{

this -> first\_name;

this → last\_name;

}

So our constructor is the blank one that will then be used with the subsequent command to fill the fields.

b.Objective-Cc.C# Same as above

d.Java initialize a variable as soon as we create an object.

Public class example {

public example (String name){

userName = name;

}

}

class names {

public static void main (String[] args) {

example exampleName = new example (“<any name>”)

example ExampleName2 = new example (“<any different name > “)

exampleName.example();

}

}

Similar to C++, except:–All user-defined types are classes–All objects are allocated from the heap and accessed through reference variables–Individual entities in classes have access control modifiers (private or public), rather than clauses–Java has a second scoping mechanism, package scope, which can be used in place of friends•All entities in all classes in a package that do not have access control modifiers are visible throughout the package