**Chapter 1 Data Abstraction: The Walls**

**Multiple Choice Questions**

1. The specifications of an ADT’s operations indicate \_\_\_\_\_\_.
   1. what the operations do
   2. how to implement the operations
   3. how to store the data in the ADT
   4. how to carry out the operations

Answer: a.

1. Data structures are part of an ADT’s \_\_\_\_\_\_.
   1. definition
   2. implementation
   3. specifications
   4. usage

Answer: b.

1. A(n) \_\_\_\_\_\_ allows two modules to communicate with each other.
   1. data structure
   2. axiom
   3. interface
   4. client

Answer: c.

1. An ADT’s \_\_\_\_\_\_ govern(s) what its operations are and what they do.
   1. specifications
   2. implementation
   3. documentation
   4. data structure

Answer: a.

1. Which element is the head of the following list: John, Kate, Fred, Mark, Jon, Adam, Drew?
   1. John
   2. Mark
   3. Drew
   4. Adam

Answer: a.

1. In the following list:

John, Kate, Fred, Mark, Jon, Adam, Drew

which element is the tail of the list?

* 1. John
  2. Mark
  3. Drew
  4. Adam

Answer: c.

1. In the following list:

John, Kate, Fred, Mark, Jon, Adam, Drew

which element does not have a predecessor?

* 1. John
  2. Mark
  3. Drew
  4. Kate

Answer: a.

1. The items in the ADT list are referenced by \_\_\_\_\_\_.
   1. name
   2. value
   3. position number
   4. position name

Answer: c.

1. The insertion operation of the ADT list can insert new items \_\_\_\_\_\_.
   1. only at the front of the list
   2. only at the end of the list
   3. only in the middle of the list
   4. into any position of the list

Answer: d.

1. In the ADT list, when an item is deleted from position *i* of the list, \_\_\_\_\_\_.
   1. the position of all items is decreased by 1
   2. the position of each item that was at a position smaller than *i* is decreased by 1
   3. the position of each item that was at a position greater than *i* is decreased by 1
   4. the position of each item that was at a position smaller than *i* is increased by 1 while the position of each item that was at a position greater than *i* is decreased by 1

Answer: c.

1. In the ADT list, when an item is inserted into position *i* of the list, \_\_\_\_\_\_.
   1. the position of all items is increased by 1
   2. the position of each item that was at a position smaller than *i* is increased by 1
   3. the position of each item that was at a position greater than *i* is increased by 1
   4. the position of each item that was at a position smaller than *i* is decreased by 1 while the position of each item that was at a position greater than *i* is increased by 1

Answer: c.

1. A client program depends solely on the \_\_\_\_\_\_ of the ADT.
   1. data members
   2. structure
   3. implementation
   4. behavior

Answer: d.

1. Which of the following operations of the ADT list changes the list?
   1. remove
   2. isEmpty
   3. size
   4. get

Answer: a.

1. The ADT sorted list inserts and deletes items by their \_\_\_\_\_\_.
   1. name
   2. value
   3. position name
   4. position number

Answer: b.

1. A(n) \_\_\_\_\_\_ can be used to precisely specify the behavior of each of the operations of an ADT.
   1. exception
   2. data structure
   3. axiom
   4. client

Answer: c.

1. Object-oriented programming views a program as \_\_\_\_\_\_.
   1. a sequence of actions
   2. a collection of classes
   3. a group of methods
   4. an interaction among objects

Answer: d.

1. An ADT’s operations are known as its \_\_\_\_\_\_.
   1. axioms
   2. methods
   3. variables
   4. interfaces

Answer: b.

1. Encapsulation combines an ADT’s data with its operations to form a(n) \_\_\_\_\_\_.
   1. exception
   2. method
   3. object
   4. variable

Answer: c.

1. A(n) \_\_\_\_\_\_ is a C++ construct that enables a programmer to define a new data type.
   1. class
   2. method
   3. data field
   4. object

Answer: a.

1. A C++ class contains data members and \_\_\_\_\_\_.
   1. clients
   2. interfaces
   3. methods
   4. data structures

Answer: c.

1. A(n) \_\_\_\_\_\_ is an instance of a class.
   1. method
   2. data field
   3. interface
   4. object

Answer: d.

1. Which of the following is true about a constructor in C++?
   1. all constructors have a return type of void
   2. a constructor cannot have parameters
   3. a constructor has the same name as the class
   4. a class can only have a single constructor

Answer: c.

1. A(n) \_\_\_\_\_\_ is a class that inherits the members of another class.
   1. base class
   2. superclass
   3. abstract class
   4. subclass

Answer: d.

1. Which of the following is true about a destructor in C++?
   1. a class can have several destructors
   2. the compiler will generate a destructor if the programmer does not provide one
   3. a programmer must provide a destructor for every class
   4. a destructor destroys all instances of a class

Answer: b.

1. In C++, each class definition should be placed in a(n) \_\_\_\_\_\_.
   1. implementation file
   2. header file
   3. namespace
   4. package

Answer: b.

1. C++ provides a mechanism for logically grouping class definitions and other declarations into a common declarative region known as a(n) \_\_\_\_\_\_.
   1. namespace
   2. interface
   3. header
   4. package

Answer: a.

1. When defining class A so that any of its instances can invoke any of the publicly defined methods of class B, the keyword \_\_\_\_\_\_ precedes the name of class B.
   1. public
   2. global
   3. static
   4. final

Answer: a.

1. A function can indicate that an error has occurred by \_\_\_\_\_\_ an exception.
   1. throwing
   2. catching
   3. implementing
   4. declaring

Answer: a.

1. To \_\_\_\_\_\_ an exception means to deal with the error condition.
   1. declare
   2. catch
   3. implement
   4. try

Answer: b.

1. Which of the following is true about runtime exceptions?
   1. they must be handled locally
   2. they must be explicitly thrown from the method
   3. they are used in situations where the method has encountered a serious problem
   4. they can often be prevented by fail-safe programming

Answer: d.

**True/False Questions**

1. According to the principle of information hiding, a module should be completely isolated from other modules.

Answer: False.

1. Data structures are part of an ADT’s implementation.

Answer: True.

1. An abstract data type is another name for a data structure.

Answer: False.

1. The head of a list does not have a successor.

Answer: False.

1. All the items in a list must be of the same data type.

Answer: True.

1. An application can use the operations of an ADT without knowing how the ADT is implemented.

Answer: True.

1. By default, all members in a class are public.

Answer: False.

1. A try block can only have one catch block associated with it.

Answer: False.

1. In C++, the scope resolution operator is a colon (:).

Answer: False.

1. Any instance of a subclass can be used in a program anywhere that an instance of the superclass can be used.

Answer: True.

**Short Answer Questions**

1. What are some of the benefits of modularity?

Answer: A modular problem is easier to write, read, and modify. Modularity also isolates errors and eliminates redundancies.

1. What is functional abstraction?

Answer: Functional abstraction is a design principle that separates the purpose and use of a module from its implementation.

1. What is information hiding?

Answer: Information hiding is a process that hides certain implementation details within a module and makes them inaccessible from outside the module.

1. What are the three types of operations on a data collection?

Answer: The three types of operations on data are:

* Add data to a data collection.
* Remove data from a data collection.
* Ask questions about the data in a data collection.

1. What is data abstraction?

Answer: Data abstraction is a design principle that separates the operations performed on a collection of data from the implementation of the operations.

1. What is an abstract data type (ADT)?

Answer: An ADT is a collection of data together with a set of operations on that data.

1. What is a data structure?

Answer: A data structure is a construct that is defined within a programming language to store a collection of data.

1. What is an axiom?

Answer: An axiom is a mathematical rule.

1. What is a constructor?

Answer: A constructor is a method that creates and initializes new instances of a class.

1. When and how are initializers used?

Answer: Initializers are used within a constructor’s implementation to set data members to initial values. Each initializer uses a functional notation that consists of a data member name followed by its initial value enclosed in parentheses. A colon precedes the first (or only) initializer, and multiple initializers are separated by commas.

1. Under what circumstance is a compiler-generated default constructor created?

Answer: A compiler-generated default constructor is created when the programmer does not include any constructor in a class.

1. Define the client of a class?

Answer: The client of a class is a program or module that uses the class.

1. What is a superclass?

Answer: A superclass is a class from which another class is derived.

1. What is the purpose of an exception?

Answer: Exceptions are used for indicating errors during the execution of a program.

1. What is the difference between the physical size of an array and its logical size?

Answer: The physical size of an array is the number of locations in the array. The logical size of an array is the number of items currently in the array.