

University of Utah

School of Computing

CS 1410 / CS 2000

Study Notes

December 10, 2011

This study guide is designed to help you prepare and study the appropriate material for the final exam. For the multiple choice questions, sample questions from previous exams are given (without answers). Additional study topics are given for additional questions. For the programming problems, questions similar to the problems below will appear on the exam. In studying to find the answers, you will learn (or make sure you know) the material that you need for the exam.

Caution: I have not had time to fully proofread this practice exam. Some questions may have typos, others may have a bit of Java in them, and others may have no correct answer. I'll make sure the actual exam questions are more robustly checked.

The final exam is a closed book/notes exam and you will have 120 minutes to complete it. No electronic devices, notes, textbooks, papers, partners, or other references will be allowed. (Bring only pencils and/or pens.) The exam will consist of about 20 multiple choice questions and two or three programming problems.

Students will write exam answers on answer sheets. Illegible handwriting will be marked incorrect, so write neatly. (No overlapping or partially erased answers.) Students will be allowed to ask for clarifications during the exam.

The exam is Thursday, December 15 at 1:00 PM in our classroom. If you need to reschedule the exam, you may take it *early*, but not late. Absolutely no late exams will be given! Email me at my personal address to reschedule.

- (1) Consider the expression `(int) 42.6`, what value and type will be the result?
- (A) A `double` equal to 42
 - (B) A `double` equal to 43
 - (C) An `int` equal to 42
 - (D) An `int` equal to 43
- (2) Consider the statement `double d = a.compute(x)`; Which one of the following **cannot** be deduced from the statement:
- (A) The `compute` function exists in `a`.
 - (B) The `compute` function takes one parameter.
 - (C) The `compute` function is not void.
 - (D) The `compute` function returns a `double` value.
- (3) Consider the looping statement `for (int i = 0; i < 10; i = i * 2)` Assuming the body of the loop does not break the loop or change `i`, how many times would the body of the loop be executed?
- (A) 0
 - (B) 3
 - (C) 7
 - (D) 10
 - (E) ∞
- (4) Consider the looping statement `for (int i = x; i <= 10; i = i + 1)` Assuming the body of the loop does not break the loop or change `i`, how many times would the body of the loop be executed?
- (A) 0
 - (B) `x`
 - (C) `10 - x`
 - (D) `11 - x`
 - (E) ∞
- (5) Consider the Boolean expression `isMoon() && (isBlue() || isFull())` Under what condition(s) will `isFull()` **not** be called when this expression is evaluated? (Select all that apply.)
- (A) `isMoon()` returns `true`
 - (B) `isMoon()` returns `false`
 - (C) `isBlue()` returns `true`
 - (D) `isBlue()` returns `false`
- (6) Which one of the following is the best *big-oh* estimate of the complexity of $(2t \lg t) + (4t) + (3\sqrt{t}) + 6000$?
- (A) $O(6000)$
 - (B) $O(t)$
 - (C) $O(t \lg t)$
 - (D) $O(\sqrt{t})$

- (7) Which of the following best completes the definition of Big-O? A function $f(n)$ is said to be $O(g(n))$ if for some non-negative constants C and K if:
- (A) $f(n) \geq C g(n)$ for all $n \geq K$
 - (B) $f(n) \geq C g(n)$ for all $n \leq K$
 - (C) $f(n) = C g(n)$ for all $n = K$
 - (D) $f(n) \leq C g(n)$ for all $n \geq K$
 - (E) $f(n) \leq C g(n)$ for all $n \leq K$
- (8) Assume an object is named `alpha` and it is of type `Greek`. What will happen when it is passed to a function that is declared like this:
- ```
int doTask(Greek a)
```
- (A) A pointer to `alpha` is stored in `a`.
  - (B) The variable `a` will refer to the same object as the object in `alpha`.
  - (C) A `Greek` object is created in `a` that gets an exact copy of `alpha`.
  - (D) The variable `a` is set to the keyword `this`.
- (9) Which one of the following is **false**:
- (A) A subclass inherits functions and variable declarations from a superclass.
  - (B) A subclass object pointer (of some superclass) may be used any place the superclass object pointer is expected.
  - (C) A superclass may override virtual functions written in the subclass.
  - (D) A subclass may have functions that are not declared in the superclass.
- (10) Select the best definition for polymorphism.
- (A) Data and functions are grouped together in a class.
  - (B) One class may extend, or specialize, another class.
  - (C) The user must not rely on the implementation details of a class.
  - (D) The behavior of an object is determined by its content, not its context.
  - (E) If class Alpha extends class Gamma, Alpha *isa* Gamma.
- (11) In C++, which one of the following is required for polymorphic behavior?
- (A) Global functions
  - (B) Destructors
  - (C) Function overloading
  - (D) Function overriding

- (12) Consider the following four statements that make use of inheritance and polymorphism. Assume classes `Alpha` and `Beta` are subclasses of class `Gamma`.

```
Alpha *a = new Alpha ();
Beta *b = new Beta ();
Gamma *g = new Gamma ();
g = b;
// Next (fifth) statement goes here.
```

Consider the fourth statement `g = b;`. Which of the following best describes the effect of this statement?

- (A) `g` gets a copy of the `Beta` object in `b`.
  - (B) `g` will point to the same object as `b`.
  - (C) A new `Gamma` object is created from `b` and is stored in `g`.
  - (D) A new `Beta` object is created from `g` and is stored in `b`.
  - (E) The code will not compile.
- (13) Which one of the following statements, when viewed as the fifth statement, would be **illegal** at compile time?

- (A) `a = new Gamma ();`
- (B) `g = new Alpha ();`
- (C) `g = a;`
- (D) `b = new Beta ();`

- (14) Which one of the following is guaranteed to be **true**?

- (A) `a.test();` is legal if `Beta` has a virtual `void test()` method.
- (B) `g.test();` is legal if `Alpha` has a virtual `void test()` method.
- (C) `a.test();` is legal if `Gamma` has a virtual `void test()` method.
- (D) `g.test();` is legal if `Beta` has a virtual `void test()` method.
- (E) `b.test();` is legal if `Alpha` has a virtual `void test()` method.

- (15) A method contract specifies preconditions and postconditions, true or false?

- (16) Testing is the least important of the stages of programming, true or false?

- (17) What will `min` and `max` contain after the following optimization loop executes? (The code demonstrates several common programming errors - be careful.)

```
int ages[] = {3, 7, 15, 21, 8, 3, 4, 56, 0, 1};

int min = ages[0];
int max = ages[0];

for(int i = 1; i < 10; i++)
{
 if(ages[i] < min)
 {
 min = ages[i];
 }

 else
 {
 max = ages[i];
 }
}
```

- (A) Min: 0, Max: 56
- (B) Min: 56, Max: 0
- (C) Min: 56, Max: 1
- (D) Min: 0, Max: 1
- (E) Min: 1, Max: 0

I recommend studying these additional topics:

- The `this` keyword
- The `NULL` keyword
- The `new` keyword
- How and why destructors are used
- The purpose of copy constructors
- Bitwise ANDing of values: `6 & 2`, etc.
- Bitwise ORing of values: `6 | 3`, etc.
- Bitwise shifting of values: `1 << 2`, etc.
- Characters, integers, and conversions
- The flow of an Arduino program
- Using Arduino Serial communications
- Timing an Arduino program
- Additional examples of reading and understanding code: Arrays, function calls, expressions and type, etc.
- Software engineering: Correct and incorrect ways to develop programs

### Programming question:

Use `if` statements, expressions, local variables, and `return` statements to complete the function below so that it satisfies the following description.

The function below should compute and return a double that corresponds to the yearly salary for an employee. The method takes two parameters, an integer that corresponds to the employee's years of experience, and a double that contains a base salary. (See the method header below.) You should calculate **and return** the salary for an employee using these rules:

- If the employee has 5 years experience or less, their salary is just the base salary.
- If the employee has more than 5 years experience, they receive a bonus of 10% of their base salary for each year of experience past 5 years.

Example: If an employee has eight years of experience, she receives her base salary plus 30% of the base salary for her three extra years of experience.

- No employee should receive more than twice the base salary. This is the maximum salary that should ever be returned.

Example: If an employee has 20 years of experience, he should get his base salary plus a bonus of 150% of the base salary as a bonus for his extra 15 years of experience. This would add up to be 250% of the base salary. Unfortunately, he can receive at most twice the base salary, so his salary is limited to 200% of the base salary.

Write statements inside the function below to complete the function. Your code will be graded for correctness, neatness, and proper syntax. A typical solution will be 6 to 10 lines of code.

```
/* Computes and returns the salary for an employee.
 *
 * The paramter 'years' contains the employee's number of years of experience.
 * The parameter 'base' contains the base salary for the employee.
 */
double calculateSalary (int years, double base)
{

}
}
```

### Programming question:

Assume you have a linked list of double values (similar to the example from class) represented by this linked list class:

```
struct Node
{
 double data;
 Node *next;
};

class LinkedListDouble
{
private:
 int count;
 Node *head, *tail;

public:

 int find (double v);

 // Other functions not shown -- assume you cannot use them
};
```

Write a complete function definition (including a small function contract) for a `find` function (shown declared above). The `find` function should search the list for the data value `v` and return the position of the value `v` in the list. (The first data value in the list is position 0, and the function should return -1 if `v` is not in the list.)

The problem will be scored on the following: Correct algorithm, correct use of pointers, correct definition of a class function, correct syntax, and neatness of communication.

You may assume the list is properly constructed, but you may not assume the list is non-empty.

A correct solution will be about 15 lines, with only 5-6 actual statements.

### Programming question:

Assume you are working on an Arduino project with a few friends – you and your friends are building a thermostat. Your friends have written these two classes that represent a way to control a furnace and an air conditioner:

```
class Cooler
{
public:
 void turnOn();
 void turnOff();
 bool isOn();
}
```

```
class Heater
{
public:
 void turnOn();
 void turnOff();
 bool isOn();
}
```

Your friends have already written these classes and the Arduino `setup` and `loop` functions, but they need help creating a `Thermostat` class.



For this problem you are to write a complete header file and `.cpp` file to declare and define a thermostat class. Your thermostat class will represent the behavior of a typical household thermostat:

- When the room temperature gets colder than some minimum temperature, your thermostat will turn on the furnace.
- When the room temperature gets warmer than some maximum temperature, your thermostat will turn on the air conditioning.
- When the temperature is between the minimum and maximum temperatures (inclusive), your thermostat will make sure the furnace and air conditioning are off.

To rephrase, your thermostat can be in exactly one of three states: cooling is on, heat is on, or everything is off. The current state is determined by the current temperature.

You won't need to do any timing, input/output code, or any other Arduino-specific code. Instead, you will just write a class that provides support functions. Your friends' code will call your class functions to set the temperatures, and your class will call pre-existing `Cooler` and `Heater` objects to turn on and off the cooler and heater.

*Continued on next page...*

*Continued from previous page...*

Your **Thermostat** class should have the following behaviors:

- Declare a few instance variables. You can determine what you need from the description of the functions below.
- Declare and define a constructor. Your constructor must require exactly five parameters: A **Cooler** object, a **Heater** object, a minimum temperature, a maximum temperature, and the current temperature (as integers).

You may assume that the **Cooler** and **Heater** objects will be valid.

You may assume that the minimum temperature is less than the maximum temperature.

Store these values as appropriate and activate the correct thermostat state. (See the next function.)

- Write a **setTempRange** function for changing the temperature range. This function must take two parameters: a minimum temperature and a maximum temperature.  
If the minimum temperature is greater than the maximum temperature, do nothing. Otherwise, store these values and activate the correct thermostat state based on the current temperature.
- Write a **setCurrentTemp** function for recording the current room temperature.  
Store the value (as appropriate for your class) and activate the correct thermostat state.
- Write a **coolerIsOn** function that will return true if the air conditioner is on, false otherwise.
- Write a **heaterIsOn** function that will return true if the furnace is on, false otherwise.

Your code will be graded on completeness, correctness, conciseness, clarity, and syntax. Comments and contracts are not required, but helpful to your grade. Illegible and confusing solutions will be marked as wrong. All global variables, user I/O, Arduino functions, arrays, new object creation, and pointers are forbidden.