Multiple Choice Questions

There are 20 multiple choice questions on the final exam. Make sure you bring an HB or darker pencil to answer these questions on a bubble sheet. To ensure you do well in this section, make sure you are able to do the following:

1. Identify the number of times (the body of) a loop will execute.
2. Know the basic milestones in the history of Computer Science since the 1940s as identified in class.
3. Understand the challenges faced by the various fields in Computer Science, as discussed in class.
4. Understand the differences and similarities between a procedural and object-oriented problem solving approach.
5. Know the various error types that can be encountered while programming and when in the coding cycle each error type is encountered.
6. Understand the difference between the coding cycle when using an interpreted language (a language with a virtual machine) and a compiled language.
7. Given a set of symbols, know the number of bits that would be needed to encode all symbols.
8. Know the commonly used encoding schemes for characters, integers and floating point numbers.
9. Able to evaluate arithmetic expression contain a mix of arithmetic operators and types of numbers.
10. Able to identify when changes to a list made inside a function are also visible outside the function.
11. Able to identify the scope of a variable. The levels of variable scope we have talked about during the semester are: local, global and class.

The following are examples of the types of multiple choice questions you can expect.

1. Given that you know the ASCII code for the character ‘A’ is 65 then what will be the ASCII code for the character ‘F’
   a. 65
   b. 66
   c. 69
   d. 70
   e. 71

2. In computer programs real numbers are stored in the floating point form. Which of the following is NOT one of the three components of a floating point number?
   a. Sign
   b. Mantissa
   c. Exponent
   d. Decimal point

3. Which of the following people were part of the team that worked on the ABC (computer)?
   a. Allan Turing
   b. Bill Gates
   c. Clifford Berry
   d. Steven Jobs
   e. None of these people were part of the team that worked on this computer.

Written Answer Questions

There is a total of 13 written answer questions on the final exam. You may use either a pen or pencil to answer these questions in the space provided in the exam booklet. To ensure you do well in this section, make sure you are able to do the following:

1. Trace a nested loop. The following are two examples of nested loops.

```
i = 0
j = 0
while (i < 5):
   while (j < 4):
      print '(' + i + ',' + j + ')  ',
      j = j + 1
   i = i + 1
   j = 0
   print '\n'
```

```
def fun1(twodlist) :
     file = open('temp.txt', 'r')
     lines = file.readlines()
     file.close()
     wordlist = []
     for line in lines :
         words = line.split()
         for word in words :
                 if word not in wordlist :
                    wordlist.append(word)
      return wordlist
```

if the file contains the following text (Coleridge's "Rime of the Ancient Mariner"), what will the function return?

I looked upon the rotting sea,
And drew my eyes away;
I looked upon the rotting deck,
And there the dead men lay.

2. Design a conditional (if statement) See the midterm for examples on this.

3. For each of the following cases, identify which is an explicit cast and which is an implicit cast.
   int(x)
   17 / 1.0
   str(15.0)
   print 15.0

4. Design a test suite given a function definition. For example, for each of the following functions, provide a set of tests that would thoroughly test the function.
   Name: moveup
   Parameters: ycoord, amount
   Pre-conditions: All parameters are numbers
   Post-conditions: The function will return a modified y-coordinate, such that the input y-coordinate is decreased by amount.

Name: is_all_caps
Parameters: str
Pre-conditions: The parameter is of type string
Post-conditions: The function will return true if all characters in str are alphabetic upper case character, false otherwise.

5. List strategies that you use when debugging and strategies that you use when testing.

6. Able to write code that uses code from other modules.

7. Write functions that parse and manipulate strings. The following are two functions that you should be able to write that parse strings.
   - Write a function that takes a single string parameter. The function should return this string but each comma (,) in the string should be replaced by a colon (:).
   - Write a function that takes two string parameter str and sub. The function should return the string str with all occurrence of sub removed.

8. Able to manipulate lists. The following are two functions that you should be able to write that manipulate lists.
   - Write a function that takes two parameters, the first a list of strings and the second a string called sub. You function should remove from the list all those string that have sub as a substring.
   - Write a function takes as parameters a list and an index. The function should move that item located at the specified index to the end of the list.

9. Able to trace exception handling code. For example, write code that will cast a variable x to an integer and print succeeded if the cast was successful and that prints 'can't convert to an integer' if the cast was not successful. Use exceptions to determine if a cast was successful or not.

10. Able to read from and write to files. For example, write a function that takes two parameters, the first a filename, the second an integer count. the lines (the number specified by second parameter) from the input file.

11. Given a problem statement, design functions that would be helpful in solving the problem. For this type of question, only the function definition (name, parameter list, data returned and possible interactions with the user). Do not provide any implementation of these function. A possible problem is as follows:

    Create a program that will read marks, expressed as letter grades, from a file. The user should be prompted for the filename. Your program will then create a bar chart based on the marks. Each bar will appear as a rectangle in Quickdraw (or some other screen). The larger the number of students have a particular grade, the higher the rectangle should be on the screen.

12. Given code that defines a class, able to identify the different components of this class. This includes the constructor, methods and instance variables. You must also be able to create an instance of this class and manipulate this instance, including call methods on the instance and passing the instance as a variable and returning an instance from a

function.

For the Time class defined below (based on the Time class defined in your text 'How to think like a Computer Scientist) write the functions

- now() which creates an instance of Time where the time is set of 11:12:13 and returns this instance.
- info(time1, time2) which takes two instance of Time (time1 and time2) and which prints the object with the earliest time to the console first and the object with the later time to the console second. Both times should printed in seconds.

```
class Time:
    def __init__(self, hours, minutes, seconds):
        self.hours = hours
        self.minutes = minutes
        self.seconds = seconds

    def after(self, time2):
        if self.hour > time2.hour:
            return True
        if self.hour < time2.hour:
            return False
        if self.minute > time2.minute:
            return True
        if self.minute < time2.minute:
            return False
        if self.second > time2.second:
            return True
        return False

    def convertToSeconds(self):
        minutes = self.hours * 60 + self.minutes
        seconds = minutes * 60 + self.seconds
        return seconds
```