University of Victoria

University of Victoria Final Examination December 2010

Course Name & No.	SENG 265 Software Development Methods
Section(s)	A01 and A02
CRN	11249 and 11250
Instructor	Michael Zastre
Duration	Three (3) hours

Name:	
Student Number: <u>V00</u>	

This exam has a total of 16 pages including this cover page.

Students must count the number of pages and report any discrepancy immediately to the Invigilator.

This exam is to be answered on the paper.

No materials or aids of any kind are permitted. This is a closed-book exam.

Ensure all cellphones are switched off. You must obtain permission from an invigilator to temporarily leave the examination room.

The total number of marks in this exam is 200.

Section A (40 marks): 20 questions

For each question in this section, **clearly circle all answers** that apply. All questions have equal weight. Partial marks are not given for incomplete answers.

Question 1. A Python list is an example of a data structure that:

a. cannot be accessed like an array.
b. may only hold strings.
c. is mutable.
d. can be appended to.
e. none of the above.

Question 2. A Python variable named `sum`:

a. may store integers.
b. is statically typed.
c. may store floats or references to strings and lists.
d. can be used before a value is assigned to it.
e. none of the above.

Question 3. To create a C-like switch statement in Python:

a. we should use the `case` statement.
b. we should use the `switch` statement
c. we should branch to a nested C script.
d. can use map on a list.
e. none of the above.

Question 4. The Python expression `[elem * 2 for elem in li]`

a. Is an example of a list comprehension.
b. Is a tuple.
f. Calls `elem` two times.
g. Produces a list with the same number of elements as in `li`.
c. None of the above.

Question 5. The Python statement "`for i in range(lower, upper): print i`"

a. sets the value of "`upper`" to "`lower`".
b. sets the value of "`lower`" to "`upper`".
c. calls the function "`range`" which returns the string "`abcde`" when lower is "`a`" and upper is "`e`".
d. will never compile because the "`for`" operator should be replaced with a "`while`" when "`range`" appears in an expression.
e. none of the above.

Question 6. Consider a Python function written as

```
def myfun(a, b, c):
    return a - b
```

a. It may be called using the expression `myfun(c=43, b=1, a=2)`
b. It will result in a Python interpreter generating a warning since the parameter `c` is not used.
c. It is a suitable definition for a method belonging to some class.
d. The parameter `c` can be a value of any type except list.
e. None of the above.

Question 7. The C string function "`strncpy(foo, bar, baz)`":

a. ensures the null character is always copied into the new string.
b. takes the string in "`bar`" and copies it to "`foo`".
c. automatically allocates memory for the new string.
d. is safer to use than "`strcpy`".
e. none of the above.

Question 8. Dynamic memory associated with a variable "`char *bar;`" in function "`foo()`":

a. may be allocated using `malloc()` or a related function.
b. may be deallocated using the `delete()` operation.
c. will be inaccessible after `foo()` returns if "`bar`" itself a local variable.
d. is located on the stack.
e. none of the above.

Question 9. The heap:

a. contains addresses that are the only legal values to be stored in C pointers.
b. is the usual storage place for anonymous variables in C.
c. is used to store variables declared locally (i.e., block scope in C).
d. is the location of garbage-collected variables in Python.
e. none of the above.

Question 10. In order for an array to grow in C we:

a. must repeatedly redeclare it with a new size.
b. must create a linked-list of arrays.
c. can use malloc to obtain more memory if the array dimensions are bound statically.
d. can use realloc to obtain more memory if the array was originally allocated in heap.
e. none of the above.

Question 11. Pipes and input/output redirection are important tools available via the UNIX shell. Which of these UNIX commands properly ensures the output of `step_a.sh` is piped into `process_b.py`, and that `process_b.py`'s output is saved in `/tmp/capture.txt`? Assume both of the scripts/programs are executable. (Remember to circle all that apply.)

```
a. cat ./options.txt | step_a.sh | process_b.py > /tmp/capture.txt
b. cat ./options.txt > step_a.sh > process_b.py > /tmp/capture.txt
c. cat ./options.txt | step_a.sh | process_b.py | /tmp/capture.txt <
d. cat ./options.txt | step_a.sh | process_b.py  || /tmp/capture.txt
```
e. None of the above.

Question 12. Python regular expressions:

a. can be used to search a list of tuples for a pattern tuple.
b. are used for regular arithmetic.
c. are conditional expressions designed to cause a sequence of function calls.
d. are only useable when passed to a function as a function.
e. none of the above.

Question 13. The UNIX "`history`" command:

a. displays a list of all previous logins to your account.
b. displays a history of all files created or destroyed in your account.
c. displays a history of errors and segmentation faults that occurred in your shell.
d. displays a list of recent commands used in your shell.
e. none of the above.

Question 14. After executing "`svn add blarg.c`", the user who issued this command:

a. knows the file named "`blarg.c`" is now tracked by the repository.
b. knows the file named "`blarg.c`" has been committed to the repository.
c. has created a new file in their working module.
d. has taken a step that may allow another user to access the contents of "`blarg.c`".
e. none of the above.

Question 15. The command "`svn checkout http://somesite.org/somemodule`":

a. can only be performed once for "`somemodule`".
b. will return an error if some other user has already checked out "`somemodule`".
c. creates possibly many subdirectories.
d. is the opposite of "`svn checkin http://foo/somemodule`".
e. none of the above.

Question 16. A bug is:

a. evidence the programmer might misunderstand the meaning of their code.
b. can always be found using "`printf`" statements in C code.
c. best found using a methodical, scientific approach.
d. an error caused by the compiler.
e. none of the above.

Question 17. "Black box" testing is the same as:

a. "functional" testing.
b. testing of Python scripts.
c. "structural" testing.
d. "blue box" testing (i.e., for checking the correctness of garbage-collector implementations).
e. none of the above.

Question 18. The `unittest` framework in Python:

a. can be used to associate test cases with class files we write in Python.
b. requires special scripting support from Unix to run the tests.
c. has methods for setting up and tearing down data structures needed for tests.
d. automatically executes the tests we have written.
e. none of the above.

Question 19. Manual testing techniques (such as Fagan inspections):

a. assume the participation of very experienced programmers as inspectors of the code being tested.
b. assume the participation of the programmer who wrote the code, who then defends that code as does a defendant at a court trial.
c. assume the existence of a checklist of well-known faults.
d. assume the existence of a moderator who organizes the inspection and chairs the inspection session.
e. None of the above.

Question 20. A "makefile":

a. is always a shell script.
b. will only work as a program if the first line is "`import unittest`".
c. requires a special "`make`" editor in order to be modified.
d. makes files for all kinds of filetypes needed by UNIX file systems.
e. none of the above.

Section B: Python (50 marks)

Question 21 (weight 10). Consider the following Python code, named "`mystery.py`":

```
#!/usr/bin/python
import sys

def main():
    (aaa, bbb, ccc, zzz) = (0, 0, 0, 0)

    ddd = sys.stdin.readlines()

    while (ddd != []):
        a_ddd = ddd[0]
        ccc = ccc + 1
        bbb = bbb + len(a_ddd)
        a_ddd = a_ddd.strip()
        words = a_ddd.split()
        aaa = len(words) + aaa
        ddd = ddd[1:]
        zzz = zzz + len([w for w in words if len(w) > 5])
        print bbb, aaa, ccc, zzz

if __name__ == "__main__":
    main()
```

and the following input, named "`in01.txt`":

```
shall i compare thee to
a summer's day? thou art more lovely and
more temperate. rough winds
do shake the darling buds of may...
```

There are no trailing spaces or tabs at the end of input lines. Write on the lines below the output that results when the input is redirected into the Python program (i.e., "`cat in01.txt | ./mystery.py`").

Question 22. Weight 30

A text file contains a sequence of x & y coordinates. These represent the endpoints of line segments (i.e., in example 2, the first line segment is from [3,5] to [10,3], the next segment is from [10,3] to [2,1], etc.).

Example file 1:
```
0 0
0 5
5 5
5 0
0 0
```

Example file 2:
```
3 5
10 3
2 1
```

Every such file contains information for at least one line segment. You may not assume a maximum number of such line segments.

(a) Write a Python function to compute and return the straight-line distance between a pair of coordinates [x1,y1], [x2,y2]:

$$dist = \sqrt{(y1 - y2)^2 + (x1 - x2)^2}$$

The exponentiation operator in Python denoted by "**". You can obtain the square root of a number by taking it to the power of 0.5. (Some marks will be given for the quality of your answer.)

(b) Write a Python script that uses the function from (a) to print the distance covered by all segments in a text file. Assume all input is from stdin. The value printed for example 1 would be 20; that for example 2 would be 15.526. (Some marks will be given for the quality of your answer.)

Question 23: Weight 10

Python strings and tuples are referred to as *immutable*, and lists are referred to as *mutable*. What is the difference between immutable and mutable variables, and what are the consequences for the Python code that you write? Provide examples.

Section C: C programming (40 marks)

Question 24. Weight 15

Consider the following C program (with line numbers provided for reference):

```
 1   #include <stdio.h>
 2   #include <stdlib.h>

 3   int a = 1;
 4   int b = 2;

 5   void function01(int x, float b)
 6   {
 7           a = a + x + (int)b;
 8   }

 9   void function02(float a, int y)
10   {
11           b = b - (int)a - y;
12   }

13   int function03(int m, int y)
14   {
15           return (a*m) + (b*y);
16   }

17   int main()
18   {
19           int a = 15;
20           function01(1, 2.6);
21           function02(3.1, 2);
22           printf("Value of a: %d; Value of b: %d\n", a, b);
23           printf("function03(2,3) == %d\n", function03(2, 3));
24           exit(0);
25   }
```

(a) Which variables have *program scope*? You may distinguish variables having the same name by including the line of their declaration.

(b) Which variables (with their types) are local to function01? in function02? in function03?

(c) For each line of the main function, write the values of the program-scope variables *after* executing that line. Note that the conversion `(int)var` truncates off fractional portion of a floating point (e.g., `(int)2.9` converts to 2). Clearly write the values beside the code line.

(d) Print the exact output of the program.

Question 25. Weight 25

Consider an array of random integers:

```
int[] arr = { 25, 20, 94, 42, 39, 38, 23, 53, 53, 86, 96,
     31, 73, 65, 5, 45, 80, 65, 81, 82, 0 };
```

The final "0" indicates that there are no more numbers to follow in the array.

Write a C function `compute_stats` accepting such an array as a parameter and returning:

- the minimum value
- the maximum value
- the average value
- the number of elements in the array

More precisely, these values must be returned in a dynamically-allocated structure, and this structure is to be of your own design. You must provide all code for the function and for the structure declaration. (Some marks will be given for the quality of your solution.)

Section D: Testing and debugging (16 marks)

Question 26. Weight 16

(a) Give three examples of *boundary conditions.*

(b) What is the difference between *testing* and *debugging*?

(c) When would you choose *white-box testing* over *black-box testing*? Why?

(d) A fellow student has called you to ask for help debugging their version of `icalprint3` (i.e., the last assignment of the course). On some input files, it segfaults at the seventh line; on other files, every third or fourth line has a missing symbol. Suggest a strategy for finding the bug – write your answer in the form of numbered points.

Section E: Potpourri (30 marks)

Question 27. Weight 10

What is a *conflict* in a source-code control system such as Subversion? And how is it resolved?

Question 28. Weight 10

When measuring the behavior of running programs, what is the difference between `gcov` and `gprof`?

Question 29. Weight 10

What is the main idea behind mutation testing?

Section F: Regular Expressions (24 marks)

Question 30. Weight 24

Consider the following table of regular-expression patterns where each pattern in a row is identified by a letter:

A	`"^[yz]+"`
B	`"0x[0-9a-f]?"`
C	`"(\w+\s){3}"`
D	`"^.*!$"`
E	`"(\/\w+)+"`
F	`"^y[z]*y$"`
G	`"(see)(run)(spot)(how) | (spot)(does) | (run)(runs)"`
H	`"[0-789]{3,5}"`
I	`"dogs|cats"`
J	`"\d\w\w\w\d?"`
K	`"d{2,5}"`
L	`"(see|run|spot|how) (spot|does) (run|runs)"`

Now consider the following table where each set of strings in a row is identified by a number:

1	`"yy", "yzy", "yzzy"`	
2	`"puce green blue", "blue yellow", "orange violet beige"`	
3	`"z", "zyz", "yyyz", "yzzz"`	
4	`"there goes the neighbourhood"`	
5	`"/home/zastre", "/usr/bin/gcc", "/var/bgates", "/tmp"`	
6	`"see spot run", "run spot run", "spot does run", "how spot runs`	
7	`"34z6", "7h9qq9", "12ha", "00x0"`	
8	`"Econ 300", "MUS 100B", "grs 100", "HINF 301"`	
9	`"0xffff", "0x30", "0x0", "0x12911"`	
10	`"123", "456", "708"`	

In each blank box beside a set of strings, write down the letter(s) corresponding to patterns matching all strings in the set. If no pattern matches, write "none" in the box. Not all patterns match to a string set.

END

This page is for the sole use of examination evaluators.

Section A	/40
Section B	/50
Section C	/40
Section D	/16
Section E	/30
Section F	/24
Total	**/200**