PLEASE HAND IN

PLEASE HAND IN

UNIVERSITY OF TORONTO
Faculty of Arts and Science

APRIL 2010 EXAMINATIONS

CSC 108 H1S
Instructors: Horton

Duration — 3 hours

Examination Aids: None

Student Number: |_|_|_|_|_|_|_|_|_|

Family Name(s): _____

Given Name(s): _____

---

*Do **not** turn this page until you have received the signal to start.*
In the meantime, please read the instructions below *carefully.*

---

This final examination paper consists of 10 questions on 20 pages (including this one). *When you receive the signal to start, please make sure that your copy of the final examination is complete.*
Comments and docstrings are not required except where indicated, although they may help us mark your answers. They may also get you part marks if you can't figure out how to write the code.
You do not need to put `import` statements in your answers.
You may not use `break` or `continue` on this exam.
If you use any space for rough work, indicate clearly what you want marked.
Assume all input is valid unless otherwise indicated; there is no need to error-check.

| | |
|---|---|
| # 1: | _____/ 7 |
| # 2: | _____/ 6 |
| # 3: | _____/ 6 |
| # 4: | _____/12 |
| # 5: | _____/ 6 |
| # 6: | _____/18 |
| # 7: | _____/ 7 |
| # 8: | _____/ 6 |
| # 9: | _____/10 |
| # 10: | _____/ 8 |
| TOTAL: | _____/86 |



*Good Luck!*

CONT'D...

## Question 1. [7 MARKS]

**Part (a)** [3 MARKS] Write the following function, according to its docstring:

```
def flip_lists(L):
    '''L is a list of 2-element lists. Reverse the order of the elements in
    the sublists of L.'''
```

**Part (b)** [4 MARKS] Write the following function, according to its docstring:

```
def valid_sizes(L, a, b):
    '''L is a list of lists; a and b are ints; a <= b. Return True iff each
    sublist has size between a and b inclusive.'''
```

## Question 2. [6 MARKS]

Write the following function, according to its docstring:

```
def interleave(s1, s2):
    '''Return a new string that contains the characters of string s1 and
    string s2 "interleaved" so that it contains the first character of s1, the
    first character of s2, the second character of s1, the second character of
    s2, etc.  If one string is longer than the other, the "extra" characters
    are added on at the end.  For example, interleave("ab", "12345") returns
    "a1b2345".'''
```

## Question 3. [6 MARKS]

The left-hand column in the table below shows a series of code fragments to be interpreted by the Python shell. For each, show the expected output in the right-hand column; if it would generate an error say so, and give the reason why.

**Hint:** Use the memory model to predict what will happen. The next page is provided for your rough work.

| Code | Output (or "error" plus reason) |
|---|---|
| `x = (1, 2, 3)`<br>`for item in x:`<br>`  item = item + 1`<br>`print x` | |
| `x = (1, 2, 3)`<br>`for i in range(len(x)):`<br>`  x[i] = x[i] + 1`<br>`print x` | |
| `x = ([1, 2], [], [8, 9, 3])`<br>`for item in x:`<br>`  item.append(5)`<br>`print x` | |
| `x = [6, 7, 8]`<br>`for item in x:`<br>`  item = item + 1`<br>`print x` | |
| `x = [6, 7, 8]`<br>`for i in range(len(x)):`<br>`  x[i] = x[i] + 1`<br>`print x` | |
| `x = [[1, 2], [], [8, 9, 3]]`<br>`for item in x:`<br>`  item.append(5)`<br>`print x` | |

*Use the space below for rough work. This page will not be marked.*

## Question 4. [12 MARKS]

Below are several functions that attempt to return True iff the given string has two characters in a row that are the same. For each version, indicate whether or not the function works. If it does not, write a call to the function that demonstrates one case where it fails (*i.e.*, the return value is incorrect or the function generates an error), and state what happens.

### Part (a) [3 MARKS]

```
def doubles1(s):
    answer = False
    previous = s[0]
    for char in s:
        if char == previous:
            answer = True
        previous = char
    return answer
```

Does the function work (circle one)?: works doesn't work

If it doesn't work, a call to the function that demonstrates this:

and the outcome of that call (circle one): incorrect return value error

### Part (b) [3 MARKS]

```
def doubles2(s):
    if s == "":
        return False
    else:
        answer = False
        previous = s[0]
        for char in s:
            if char == previous:
                answer = True
            previous = char
        return answer
```

Does the function work (circle one)?: works doesn't work

If it doesn't work, a call to the function that demonstrates this:

and the outcome of that call (circle one): incorrect return value error

## Part (c) [3 MARKS]

```
def doubles3(s):
    answer = False
    if s == "":
        return False
    else:
        answer = False
        previous = s[0]
        i = 1
        while i < len(s):
            if s[i] == previous:
                answer = True
            previous = s[i]
            i = i + 1
        return answer
```

Does the function work (circle one)?: works doesn't work

If it doesn't work, a call to the function that demonstrates this:

and the outcome of that call (circle one): incorrect return value error

## Part (d) [3 MARKS]

```
def doubles4(s):
    answer = False
    i = 0
    while i < len(s):
        if s[i] == s[i + 1]:
            answer = True
        i = i + 1
    return answer
```

Does the function work (circle one)?: works doesn't work

If it doesn't work, a call to the function that demonstrates this:

and the outcome of that call (circle one): incorrect return value error

## Question 5. [6 MARKS]

Suppose we record information about students in a dictionary, where each key is student number (an int), and its value is a dictionary of information about that student. These inner dictionaries are structured like this: `{"gpa": 3.95, "name": "Lucia", "campus": "stg"}`. Suppose also that we have put the students into groups for a project, and that we keep track of the groups in a list of lists, where each inner list contains the student numbers of the members of one group.

Write the following function according to its docstring:

```
def describe_groups(groups, students):
    '''For each group, print the group size, average GPA, and the number of
    stg students. groups is a list, and students is a dict, as described
    above.'''
```

## Question 6. [18 MARKS]

Students in my other course, csc343, get 3 grace days (sorry about that!). For each assignment, I have a file with the grades and information about grace day ussage. Here is the beginning of the Assignment 1 file:

```
* CSC343H1S 20101: Assignment 1 marks for Horton's section
A1 / 75
A1grace / 3

123456789,Simpson,Homer,65,0
234567890,Kent,Clark,36,1
345678901,Horton,Diane,47,1
456789012,Flintstone,Wilma,70,0
567890123,Rubble,Barney,55,2
```

The first several lines are a header with information that is irrelevant to this question and will therefore need to be skipped over. The number of lines in the header is unknown; it ends with a blank line. The remaining lines have the following components, separated by commas:

- a nine-digit student number
- family name, which might include blanks
- first names, which might include blanks
- assignment grade (between 0 and 100)
- number of grade days used (between 0 and 3)

The list of students may not be exactly the same from one assignment file to another, since students can add and drop courses.

Your job is to write a program that will read my grades files `a1.txt`, `a2.txt`, and `a3.txt` and print the student number (one per line) of every student who has used more than three grace days in total. Your program will build and use a dictionary where each key is a student number, and its value is the number of grace days used by that student. The dictionary will start out empty, and each file you read will update it.

You will write three helper functions and then the main block. Wherever appropriate, you should call functions from earlier parts of the question in your solutions for later parts, even if you haven't written them. Just assume they work.

**Part (a)** [3 MARKS] Write this helper function, according to its docstring.

```
def skip_header(r):
    '''r is an open reader for a grade file (in which nothing has been read
    yet). Skip past the header and the blank line, so that the next line to be
    read is the first line of student data.'''
```

**Part (b)** [7 MARKS] Write this helper function, according to its docstring.

```
def add_grace(r, grace_used):
    '''r is an open reader for a grade file (in which nothing has been read
    yet). grace_used is a dict whose keys are student numbers (as strings) and
    whose values are ints indicating the number of grace days used by a
    student. Read the grace day usage information in r and update grace_used
    for each student encountered in r. If a student is not in the dict, add
    him or her.'''
```

**Part (c)** [3 MARKS] Write this helper function, according to its docstring.

```
def print_overused(grace_used, limit):
    '''grace_used is a dict whose keys are student numbers (as strings) and
    whose values are ints indicating the number of grace days used by a
    student. Print, one per line, the student number of every student whose
    number of grace days used exceeds limit (an int).'''
```

**Part (d)** [5 MARKS] Write the main block (including its `if` statement), so that the program accomplishes what was described in the question.

## Question 7. [7 MARKS]

Below is the outline of a program for keeping track of your DVDs. Its main block will use a variable called dvds to store information about them. You have three tasks:

1. Complete the main loop so that it calls the appropriate function from the main_choices dictionary. Note that all these functions need one argument: dvds.
2. Modify the program as appropriate so that it initializes the variable dvds by loading it from a cPickle file called data.pkl, if that file exists. If not, it should initialize dvds to be an empty dictionary. Recall that you can use os.path.exists('data.pkl') to see if file data.pkl exists.
3. Modify the program as so that right before it ends, it saves the contents of variable dvds in a cPickle file called data.pkl.

You may add, delete or change existing code. If your new code won't fit in the space available, use arrows to point to where the pieces should go.

```
import easygui, cPickle, sys, os
# Definitions of functions acquire_dvd, list_dvds, lend_dvd and return_dvd
# omitted to save space.

def quit(dvds):



    sys.exit()


if __name__ == '__main__':









    main_choices = {"lend" : lend_dvd, "return" : return_dvd,
                "acquire": acquire_dvd, "list": list_dvds, "quit": quit}


    while 1:
        choice = easygui.buttonbox("Action", "DVD library", main_choices.keys())


        # Make sure the user chose something.
        if choice:
```

## Question 8. [6 MARKS]

Don't guess. There is a 1-mark **deduction** for wrong answers on this question.

### Part (a) [2 MARKS]

```
def f1(L):
    sum = 0
    for item in L:
        for i in range(1, 101):
            if item >= i:
                sum = sum + 1
    return sum
```

Let $n$ be the size of the list L passed to this function. Which of the following most accurately describes how the runtime of this function grow as $n$ grows? Circle one.

(a) It grows linearly, like n does. (b) It grows quadratically, like $n^2$ does.

(c) It grows less than linearly. (d) It grows more than quadratically.

### Part (b) [2 MARKS]

```
def f2(L):
    sum = 0
    i = 1
    while i < len(L):
        sum = sum + L[i]
        i = i * 2
    return sum
```

Let $n$ be the size of the list L passed to this function. Which of the following most accurately describes how the runtime of this function grow as $n$ grows? Circle one.

(a) It grows linearly, like n does. (b) It grows quadratically, like $n^2$ does.

(c) It grows less than linearly. (d) It grows more than quadratically.

### Part (c) [2 MARKS]

```
def f3(L):
    for i in range(len(L)):
        L[i] = L[i] + 1
    sum = 0
    for item in L:
        sum = sum + item
    return sum
```

Let $n$ be the size of the list L passed to this function. Which of the following most accurately describes how the runtime of this function grow as $n$ grows? Circle one.

(a) It grows linearly, like n does. (b) It grows quadratically, like $n^2$ does.

(c) It grows less than linearly. (d) It grows more than quadratically.

## Question 9. [10 MARKS]

Consider the following class:

```
class AcademicHistory(__builtin__.object)
 |  A student's academic history.
 |
 |  Methods defined here:
 |
 |  __cmp__(self, other)
 |      Return -1 if this student's average is lower than other's,
 |      +1 if it is higher, and 0 if they are equal.
 |
 |  __init__(self, stnum, family_name, first_name)
 |      A new academic history for the student with student number stnum
 |      (an int), and the given family_name and first_name (strs).
 |
 |  __str__(self)
 |      Return a str with the student's student number, name, course
 |      results, and average grade.
 |
 |  average(self)
 |      Return the average grade (a float) among this student's completed
 |      courses.
 |
 |  complete(self, course, grade)
 |      Record the fact that the student completed this course (a str)
 |      with this grade (an int).
```

**Part (a)** [7 MARKS]

1. Create an `AcademicHistory` for a student named Nora Kingston, and make variable `nora` refer to it. Record the fact that she completed CSC108 with a grade of 72.

2. Which method does the above code call without mentioning it by name?

3. Without calling any `AcademicHistory` method by name, print out Nora's academic history.

4. Suppose you have a second `AcademicHistory` called `balraj`. Write a loop that will record the fact that Balraj earned 100 in each of these courses: HIS150, MUS100 and POL121.

5. Write a single line of code that will cause `__str__` to be called without mentioning it by name.

**Part (b)** [3 MARKS] Complete the following function according to its docstring description.

```
def best_student(L):
    '''L is a non-empty list of AcacemicHistories. Return the student number
    of the one with the highest average.'''
```

## Question 10. [8 MARKS]

Throughout this question, assume that we are sorting lists into non-descending order.
**Do not guess.** There is a one-mark deduction for incorrect answers.

### Part (a) [2 MARKS]

Consider the following lists: `L1 = [9, 8, 7, 6, 5, 4, 3, 2, 1]` and `L2 = [8, 1, 3, 6, 9, 7, 4, 2, 5]`
Which list would cause bubblesort to do more swaps? Circle one answer.

`L1` `L2` they would both require an equal number of swaps

### Part (b) [2 MARKS]

Suppose you have a list of 100 elements. Is it possible that if you call bubblesort on it, no items will move? Circle one.

yes no

### Part (c) [2 MARKS]

Consider a list of 100 distinct numbers in order from largest to smallest, in other words, backwards to the order we want. Which sorting technique would cause items to be moved more times, if called on this list? Circle one.

selection sort insertion sort

### Part (d) [2 MARKS]

We are partly through sorting a list, and have completed 4 passes through the data. The list currently contains `[9, 12, 42, 59, 2, 99, 10, 60]` Which sorting technique are we definitely *not* using. Circle one.

selection sort insertion sort

*[Use the space below for rough work. This page will* ***not*** *be marked, unless you clearly indicate the part of your work that you want us to mark.]*

*[Use the space below for rough work. This page will* **not** *be marked, unless you clearly indicate the part of your work that you want us to mark.]*

## Short Python function/method descriptions:

```
__builtins__:
  len(x) -> integer
    Return the length of the list, tuple, dict, or string x.
  max(L) -> value
    Return the largest value in L.
  min(L) -> value
    Return the smallest value in L.
  open(name[, mode]) -> file object
    Open a file.  Legal modes are "r" (read), "w" (write), and "a" (append).
  range([start], stop, [step]) -> list of integers
    Return a list containing the integers starting with start and ending with
    stop - 1 with step specifying the amount to increment (or decrement).
    If start is not specified, the list starts at 0.  If step is not specified,
    the values are incremented by 1.
cPickle:
  dump(obj, file)
    Write an object in pickle format to the given file.
  load(file) --> object
    Load a pickle from the given file
dict:
  D[k] --> value
    Return the value associated with the key k in D.
  k in d --> boolean
    Return True if k is a key in D and False otherwise.
  D.get(k) -> value
    Return D[k] if k in D, otherwise return None.
  D.keys() -> list of keys
    Return the keys of D.
  D.values() -> list of values
    Return the values associated with the keys of D.
  D.items() -> list of (key, value) pairs
    Return the (key, value) pairs of D, as 2-tuples.
file (also called a "reader"):
  F.close()
    Close the file.
  F.read([size]) -> read at most size bytes, returned as a string.
    If the size argument is negative or omitted, read until EOF (End
    of File) is reached.
  F.readline([size]) -> next line from the file, as a string. Retain newline.
    A non-negative size argument limits the maximum number of bytes to return (an incomplete
    line may be returned then).  Return an empty string at EOF.
float:
  float(x) -> floating point number
    Convert a string or number to a floating point number, if possible.
int:
  int(x) -> integer
    Convert a string or number to an integer, if possible.  A floating point
    argument will be truncated towards zero.
list:
  x in L --> boolean
    Return True if x is in L and False otherwise.
  L.append(x)
    Append x to the end of the list L.
```

```
  L.index(value) -> integer
    Returns the lowest index of value in L.
  L.insert(index, x)
    Insert x at position index.
  L.remove(value)
    Removes the first occurrence of value from L.
  L.reverse()
    Reverse *IN PLACE*
  L.sort()
    Sorts the list in ascending order.
str:
  x in s --> boolean
    Return True if x is in s and False otherwise.
  str(x) -> string
    Convert an object into its string representation, if possible.
  S.find(sub[,i]) -> integer
    Return the lowest index in S (starting at S[i], if i is given) where the
    string sub is found or -1 if sub does not occur in S.
  S.index(sub) -> integer
    Like find but raises an exception if sub does not occur in S.
  S.isdigit() -> boolean
    Return True if all characters in S are digits and False otherwise.
  S.lower() -> string
    Return a copy of the string S converted to lowercase.
  S.lstrip([chars]) -> string
    Return a copy of the string S with leading whitespace removed.
    If chars is given and not None, remove characters in chars instead.
  S.replace(old, new) -> string
    Return a copy of string S with all occurrences of the string old replaced
    with the string new.
  S.rstrip([chars]) -> string
    Return a copy of the string S with trailing whitespace removed.
    If chars is given and not None, remove characters in chars instead.
  S.split([sep]) -> list of strings
    Return a list of the words in S, using string sep as the separator and
    any whitespace string if sep is not specified.
  S.strip() -> string
    Return a copy of S with leading and trailing whitespace removed.
  S.upper() -> string
    Return a copy of the string S converted to uppercase.
```

Total Marks = 86