CSC 108H1 F 2008 Test 3
Duration — 35 minutes
Aids allowed: none

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

Lab time or TA name: _____

Last Name: _____ **First Name:** _____

Instructor: Petersen

---

*Do **not** turn this page until you have received the signal to start.*
(Please fill out the identification section above, **write your name on the back of the test**, and read the instructions below.)
Good Luck!

This test consists of 3 questions on 8 pages (including this one). *When you receive the signal to start, please make sure that your copy is complete.*
Comments 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.
If you use any space for rough work, indicate clearly what you want marked.

1: _____/ 6

2: _____/ 6

3: _____/ 8

TOTAL: _____/20

Question 1. [6 MARKS]

Consider the function `square_sequence` below:

```
def square_sequence(s):
    '''Given a sequence s of numbers, return the same sequence except that
    each value is replaced by its square.'''

    for i in range(len(s)):
        s[i] = s[i] * s[i]
    return s
```

What will be printed after each of the following code fragments is executed? If the code fragment runs correctly, state what will be printed in the space provided. If you think the fragment produces an error, say so, and explain why it fails.

```
s1 = range(0, 15, 3)
print square_sequence(s1)
```

```
s2 = (-1, 3, -5, 7)
print square_sequence(s2)
```

```
s3 = {3: 1, 7: 0, 'a': 2}
print square_sequence(s3)
```

Question 2. [6 MARKS]

Complete the following function according to its docstring description. Assume that all coordinates `(x, y)` are within the bounds of the picture.

```
import media

def add_lines(pic, lines_dict):
    '''Add the lines specified in dictionary lines_dict to the picture pic.
    lines_dict contains coordinate tuples as keys.  The values in lines_dict
    are tuples containing a list of coordinate tuples and a colour. The lines
    specified in a key:value pair in the dictionary go from the point specified
    in the key to each of the points given in the coordinate list in the value.

    For example, given the following key:value pair from the dictionary:
        (3, 5): ([(4, 6), (2, 9)], Color(255, 255, 255))
    Add a line in pic from point (3, 5) to (4, 6) and a line from (3, 5)
    to (2, 9).  Both lines should be coloured Color(255, 255, 255).'''
```

Question 3. [8 MARKS]

Write a program that prompts the user for a Python file using `choose_file`, opens the file, creates a dictionary containing the names and parameters of all functions in the file using function `find_functions`, and then prints the dictionary. This program consists of two parts: a function, `find_functions`, and a main block at the bottom of the next page.

For this Python file:

```
def f():
    pass

def g(x, y):
    pass
```

Your program should print:

```
{'f': [], 'g': ['x', 'y']}
```

The header for `find_functions`, including a docstring describing its behaviour, is included below. Assume that all function definitions start with the string "def". Also assume that you have been given a function named "parse_func_def" (from some module called "function_parser") that can break apart a line containing a function definition. Use "parse_func_def" in `find_functions`. Here is the help information from `help(function_parser.parse_func_def)`:

```
parse_func_def(line)
    Return a tuple (func_name, func_parameters) where func_name is the name of the function
    defined in the string line and func_parameters is a list of strings representing the
    parameters of the function defined in line.
```

```
--------------------------------------------------------------------------------
import media
import function_parser

def find_functions(f):
    '''Return the dictionary containing all of the functions from the open Python file f.
    For each function, the dictionary should contain the function's name as a
    key and a list containing the names of its parameters as a value.'''
```

More space for your answer to Question 3.

```
if __name__ == "__main__":
```

[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 or string x.
  open(name[, mode]) -> file object
    Open a file.
  range([start], stop, [step]) -> list of integers
    Return a list containing the integers starting with stop 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.
dict:
  D.get(k) --> value
    Return the value associated with the key k in D.
  D.has_key(k) --> boolean
    Return True if k is a key in D and False otherwise.
  D.keys() --> list of keys
    Return the keys of D.
  D.values() --> list of values
    Return the values associated with the keys of D.
file:
  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 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.
media:
  add_line(pic, x1, y1, x2, y2, col)
    Draw a line of Color col from (x1, y1) to (x2, y2) on Picture pic.
  Color(red, green, blue) -> Color
    Define a Color with the specified red, green, and blue components.
  set_color(Pixel, Color)
    Set the RGB values of the given Pixel to those of the given Color.
str:
  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.replace(old, new) --> string
    Return a copy of string S with all occurrences of the string old replaced
    with the string new.
  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.startswith(prefix) -> bool
    Return True if S starts with the specified prefix and False otherwise.
  S.strip() --> string
    Return a copy of S with leading and trailing whitespace removed.
```

Last Name: _____ **First Name:** _____