CS 301 - Fall 2016

Instructor: Laura Hobbes LeGault

Midterm Exam 2 — 16.67%

(Last) Surname: _____ (First) Given name: _____

NetID (email): _____ @wisc.edu

IMPORTANT: Answers for Dual and Multiple Choice questions *must* be marked on a scantron. The answer marked on the scantron will be the only answer graded.

Fill in these fields (left to right) on the scantron form (use #2 pencil):

1. LAST NAME (surname) and FIRST NAME (given name), fill in bubbles
2. IDENTIFICATION NUMBER is your Campus ID number, fill in bubbles
3. Under ABC of SPECIAL CODES, write 001 (morning lecture), fill in bubbles
4. Under J of SPECIAL CODES, write A (exam version), fill in bubble 0

I certify that I will keep my answers covered and do my best to not allow my exam paper to be viewed by another student during the exam or prior to completion of their exam. I also certify that I have not viewed or in any way used another's work in completing my answers. I understand that being caught allowing another to view my work or being caught viewing another's work are both violations of this agreement and either will result in automatic failure of the course and an academic misconduct letter to the Deans Office for myself and any other individuals involved.

Signature: _____

The following exam has 24 questions and is worth a total of 88 points. You will have 50 minutes to complete the exam. **Be sure to read through every question completely.**

1. **Dual Choice** — 12 questions worth 2 points each. Choose the *best* answer.
2. **Multiple Choice** — 10 questions worth 4 points each. Choose the *best* answer.
3. **Fill-in-the-blank** — 4 blanks worth 6 points each. Be complete.

You may not use notes or books, your neighbors, or calculators or any other electronic devices on this exam. **Turn off and put away** any portable electronics now.

Disclaimer: the following are provided for your reference only, and the inclusion of information here does not guarantee it will be used on the exam.

Operator Precedence Table:

level	operator	description
higher	`( <expression> )`	grouping with parentheses
	`x[index]`	indexing
	`* / %`	multiplicative
	`+ -`	additive
↕	`< <= > >=`	relational
	`== !=`	equality
	`not`	logical not
	`and`	logical and
lower	`or`	logical or
	`= += *=`	(compound) assignment

Built-in functions:

`raw_input(p)`	Prompts the user for input using `p` and returns the user's input as a string.
`len(s)`	Return the length (the number of items) of an object.
`range(n)`	Returns a list of `n` consecutive integers beginning at `0`.
`range(a,b)`	Returns a list of consecutive integers beginning at `a` and ending before `b`.
`type(x)`	Returns the *data type* of the value stored in `x`

Constants and functions from math, string, and random modules:

`math.pi`	The mathematical constant $\pi = 3.141592...$
`w.isalpha()`	Return true if all characters in string `w` are letters, `w` not empty.
`w.isdigit()`	Return true if all characters in string `w` are numbers, `w` not empty.
`random.randint(a,b)`	Return a random integer N such that `a <= N <= b`.

List and dictionary methods:

`list.append(x)`	Add the value `x` to the end of `list`, in place.
`list.insert(i,x)`	Insert the value `x` at the `i`th index of `list`, in place.
`list.remove(x)`	Remove the first instance of the value `x` from `list`, in place.
`list.pop(i)`	Remove the value at index `i` from `list`, in place.
`dict.keys()`	Return a copy of `dict`'s list of keys.
`dict.values()`	Return a copy of `dict`'s list of values.

Dual Choice: Terminology

1. If `nums` is a list, the code `nums2 = nums` creates a _____ copy. (2)

   A. deep
   B. shallow

2. If `dict` is a dictionary, in the code (2)

```
for x in dict:
```

   `x` is a _____ in the dictionary.

   A. key
   B. value

3. Adding one to the value of an integer variable is called _____ . (2)

   A. incrementing
   B. iterating

4. If `word = "pie"`, the code `word[0] = "P"` *fails* because `word` is _____ . (2)

   A. immutable
   B. mutable

5. To add a value to the *beginning* of a list, you must _____ it. (2)

   A. append
   B. insert

6. List comprehension creates a new list using a _____ loop in brackets. (2)

   A. `while`
   B. `for`

7. An *ordered* collection of values would best be stored in a _____ . (2)

   A. list
   B. dictionary

8. If `L` is a list of 9 elements, the code `L[3]` is an example of _____ . (2)

   A. indexing
   B. slicing

True or False: Evaluating boolean expressions

9. `not "W".isalpha()` (2)

   A. True
   B. False

10. `x != "Y" or x != "N"  # no, you do not need to know the value of x` (2)

   A. True
   B. False

11. `"A" in {1:"A", 2:"B", 3:"C"}` (2)

   A. True
   B. False

12. `len( range(n) ) == n` (2)

   A. True
   B. False

Multiple Choice: Reading code

13. Which of the following types is *not* a legal dictionary **value**? (4)

   A. `str` (string)
   B. `int` (integer)
   C. `dict` (dictionary)
   D. All of the above are legal dictionary **value** types.

14. What is the *error* produced when the following code is run? (4)

```
for letter in len("parrot"):
   print (letter),
```

A. `IndexError: string index out of range`

B. `TypeError: 'int' object is not iterable`

C. `NameError: global name 'letter' is not defined`

D. This code does not cause an error.

15. What is the *type* of `x` after the following line of code is executed? (4)

```
x = input("Age:").isdigit()
```

A. `str` (string)

B. `bool` (boolean)

C. `float`

D. `int` (integer)

16. Which of the following best describes the *value* in `x` after this code executes? (4)

```
import random
x = [ random.randint(1,20) for x in range(50) if x % 2 == 0 ]
```

A. A list containing a function.

B. A list containing 50 random ints between 1 and 20.

C. A list containing 50 random ints between 1 and 19.

D. A list containing 25 random ints between 1 and 20.

17. If the following code **does not** cause an error, what must the *data type* of `x` be? (4)

```
x[0] == 4
```

A. `str` (string)

B. `dict` (dictionary)

C. `list`

D. All of the above are legal data types for `x` with this syntax.

18. Which statement most accurately explains *why* the following function does *not* succeed in removing all elements from the list it receives as an argument? (4)

```
def clear(list_to_clear):
    for element in list_to_clear:
        list_to_clear.remove(element)
```

A. The `for` loop combined with `remove()` will skip elements.

B. The function never returns the cleared list.

C. The `remove()` function requires an index, not an element.

D. `remove()` only removes the first matching element, any duplicates will remain in the list.

19. Given `x = [["one", 2.0, 3], "abc"]`, what is **not** a *data type* of `j` while the following code executes? (4)

```
for i in x:
    for j in i:
        print (j),
    print()
```

A. `int` (integer)

B. `list`

C. `float`

D. `str` (string)

20. What is the *value* in `x` after the following line of code is executed? (4)

```
x = ['a', 'b'].append('X')
```

A. `None`

B. `'X'`

C. `['a', 'b', 'X']`

D. `['X', 'a', 'b']`

21. Given that `dict` is a dictionary initialized as (4)

```
dict = {"A":1, "B":2, "C":3}
```

which of the following lines of code assigns the value `0` to the key `"B"`?

A. `dict["B"] -= 2`

B. `dict[0] = "B"`

C. `dict[1] = 0`

D. `dict.insert("B", 0)`

22. **Challenge!** What is the *value* in `x` after the following code is executed? (4)

```
def fcn1(x):
   if x == 0:
      return 0
   return x * fcn2(x-1)

def fcn2(y):
   if y == 0:
      return 1
   return y + fcn1(y-1)

x = fcn1(3)
```

A. 12

B. 9

C. 3

D. 1

Fill-in-the-blank: Writing code

Fill in the blanks to complete the functions as their docstrings indicate. Each blank is worth **6 points**, and there are a total of 4 lines.

23.

```
import random

def choose_revealed_door (prize_door, contestant_door, num_doors):
    """ After the contestant chooses a door in the Monty Hall problem,
        the host reveals a door that does not contain the prize and is
        also not the contestant's choice.  This function should return
        the INDEX (not number) of the revealed door.
    """
    host_door = 0        # initialize - host picks the first door

    while ____________________________________________________:        (6)

        host_door = random.randint(____________, ____________)        (6)
    return host_door
```

24.

```
def add_to_sorted_list (sorted_list, new_value):
    """ This function implements insertion sort, where a new value is
        added at the correct position in an already-sorted list.
        For example, the arguments
            sorted_list = [1, 5, 10]
            new_value = 3
        would produce the list [1, 3, 5, 10].
    """
    index = 0

    while ________________________ < new_value:    # find right location    (6)
        index += 1

    sorted_list.__________________________________      # add to list        (6)
```

This page intentionally left blank.

Please leave it attached to your exam.