Introduction to Computer Programming — Fall 2022 — **Problem Set**

Attempt seriously by **Monday, November 7** *before lecture*

Submit *hardcopy* on **Thursday, November 10**, *at start of class*

Your name:

Instructions

- *Do the problem set on your own.* Questions should be directed to the instructor and no one else.[1]
- Write your name where indicated above (on this page) and, then, on the top of *every* page you submit.
- When asked, provide clear, *concise* explanations.
- *Handwrite* your final solutions in **black ink**. If you have trouble writing legibly (I sympathize!) then you may *staple* one or more typed pages (printed in *black ink* on *otherwise blank paper*) to your problem set. Write drafts of your solutions first; write your final answers on the printed problem set only when ready.
- If you wish me to print this document for you, let me know *before* Monday.
- You *may*:
  - + refer to notes you have taken in class
  - + refer to lecture notes or other material I have posted for the class (if there are lecture notes that are not yet posted that you wish to see, let me know, the sooner the better)
  - + email me questions pertaining to this problem set as often as you would like.
- You may ***not***:
  - − discuss any of the problems with friends, families or strangers
  - − use the internet (other than to review class material that I have posted)
  - − run Python (or other) programs to evaluate any of the problems
  - − use a calculator.
- If you are unsure whether a resource is permitted, ask me first; *err on the side of caution.*

It *may* help to remember that the first few powers of two are:

```
1    2    4    8    16    32    64    128    256    512    1024
```

[1] If you ask me a question, I *may* direct you to attend a tutoring session at the Learning Commons to work out examples *related to, but not identical to* exercises on this problem set.

0 10 kinds of computational thinking

a. Express the binary number **10010101** in base ten. Show your work.

b. Express **298** in binary. Show your work.

c. Suppose you already knew the answer to one of the previous two exercises. Explain why you might then be able to more easily derive the answer to the other exercise.

1 the surprising truth about programming

For each of the following expressions, indicate to what the expression evaluates and explain, briefly, why that result might be surprising to someone first learning to program:

a. `47 == '47'`

b. `'Boolean' < 'algorithm'`

c. `'ABCD'[3] == 'C'`

d. Write a Boolean expression (and what it evaluates to) that illustrates yet a different way in which learning to program can be surprising. Explain your choice.

2 getting loopy

a. When implementing a REPL, does it make more sense to use a definite or an indefinite loop? Explain.

The following function is intended to return the number of occurrences of character `c` in the string `s`:

```
def count(s, c):
    counter = 0
    i = 0
    while i < len(s):
        if s[i] == c:
            counter += 1
            i += 1
    return counter
```

b. It has a bug! Explain the bug and how it illustrates one challenge specific to learning Python.

c. How does the `count` example indicate an important difference between definite and indefinite loops?

3 Pythontopia

Consider this function:

```
1  def htols(a):
2      b = 0
3      for c in a:
4          b += 1    # <---
5      return b
```

a. What can you say about the *types* of `a`, `b` and `c`? Explain.

b. Suppose `htols('Fun!')` is evaluated. Indicate the values of `b` and `c` immediately *after* each time line 4 is executed. (*Leave extra rows, if any, blank.*)

#	b	c
1		
2		
3		
4		
5		
6		

c. Write a one-sentence description of what `htols` returns.

d. We have discussed a range of desirable properties (correctness, efficiency, etc.) for a program. Along those lines, identify a way in which `htols` might be considered lacking. Be specific.

4 more fun

Consider this function:

```
1  def mune(a):
2      b = 0
3      c = ()
4      for d in a:
5          if d.isupper():
6              c += ((b, d),)
7          b += 1   # <---
8      return c
```

a. What can you say about the *types* of `a`, `b`, `c` and `d`? Explain.

b. What does `mune(())` return?

c. Suppose `mune('McFly')` is evaluated. Indicate the value of `c` each time line 7 is executed. (*Leave extra rows, if any, blank.*)

#	c
1	
2	
3	
4	
5	
6	
7	
8	

d. Write a one-sentence description of what `mune` returns.

5 thought questions

a. Describe (in a minimum of 50 words, maximum of 200) a compelling idea you have learned in this course and why you hope to learn more about it in the remaining weeks of the semester.

b. Write a response (minimum 50 words, maximum 200) to *exactly one* the following two prompts:

Arghhh! Describe what you have found most frustrating about writing Python programs.

Eureka! Describe what you have found most satisfying about writing Python programs.