NOTE: This is the first midterm from Fall, 2018. Some important differences to note between this exam and the one you will take. Our exam is taking place somewhat later in the semester, and thus will include material on lists that does not appear here.

You can use your sheet of notes but should have no electronic devices out in front of you. The various parts of each of the four problems below are largely independent of one another, which means you can solve the later parts of the problem even if you did not correctly solve the earlier parts. The exception is Problem 3, in which part (b) does depend on part (a). 1(b) is a bit tricky. Do all of your work in the blue exam book provided.

Each part of each problem is worth 10 points.

1. The function qualifying, whose code is shown in Figure 1, is a simplified version of one of those ‘can I get into college?’ decisions. (As you probably can guess, the function max returns the larger of the values x and y, and min returns the smaller of the two values.)

   ```python
   def qualifying(m,v):
       if max(m,v)>=600:
           if min(m,v)>=500:
               return True
           if m+v>=1200:
               return True
       return False
   ```

   **Figure 1. The function for Problem 1.**

   (a) What is the value returned by this function when it is called with arguments 700,400? With the arguments 650,500?
   (b) Suppose the line if m+v>=1200 is changed to elif m+v>=1200.
       This changes the behavior of the function. Give an example of a pair of values m and v for which the original function and this modified version give different results.
   (c) Write a function equivalent to qualifying that does not use any if statements. (It can be done in a single line.)
2. Consider the function `trits` whose code is shown in Figure 2. Observe that this function prints output rather than returning a value.

```python
def trits(n):
    while n>0:
        print(n%3)
        n=n//3
```

*Figure 2. The function for Problem 2*

(a) What is printed when you call `trits(2)`? `trits(16)`? `trits(-1)`?

(b) A common error in writing such code is to put statements that belong in a loop outside the loop, and vice-versa. What would be the result of calling `trits(16)` if the statement `n=n//3` were moved to the left, so that it started in the same column as the word `while`?

(c) Another common error is to use the wrong kind of division. What would be the result of calling `trits(16)` if the statement `n=n//3` were replaced by `n=n/3`?

3. Consider the function `design` in Figure 3. Once again, this function prints output rather than returning a value. (Recall that if the arguments to the `print` function include `end=' '`, then the output does not advance to the next line, and the next thing to be printed will appear immediately to the right of the output. The statement `print()` advances to the next line.)

```python
def design(s):
    m=(len(s)+1)//2
    for row in range(m):
        for column in range(m):
            print(s[row+column],end='')
        print()
```

*Figure 3. The code for Problem 3.*

(a) What is the output that results when you call `design('ABCDE')`?

(b) You should now have an idea of what this function prints in general. (You might observe that the answer is slightly different, depending on whether the argument is a string of odd or even length.) Write an equivalent function that has only a single for statement, instead of nested for statements, by using string slices.
4. A standard problem in introductory calculus classes is to find the largest possible volume $V$ in cubic inches of a cylindrical soup can made from $S$ square inches of metal. But you don’t have to do any calculus here! The answer is

$$V = 2\pi \left( \frac{S}{6\pi} \right)^{3/2}$$

(a) The code shown in Figure 4 is supposed to perform this calculation and return the resulting volume. However there are two errors in the code, both of which cause the function to return an incorrect answer. Correct these errors. (The value of $\pi$ is obtained by importing the math library, which is why it appears as `math.pi` in this code.)

(b) Write a main program that uses this function to print a table of the maximum volumes for $S=10, 20, \ldots, 100$. You don’t have to worry about pretty formatting or displaying a reasonable number of decimal places. Just make sure your table contains two numbers—the values of $S$ and $V$—on each line. This should not be long—not more than four or five lines of code at most.

```python
def maximal_volume(s):
    return 2*math.pi*(s/6*math.pi)**3/2
```

*Figure 4. Code for Problem 4.*