

Show Your Work! The point values for each problem are in square brackets. There are 35 points possible.

1. Bob has written a Python function called `weeb`. It has the following header:

```
def weeb(x):
```

Is the following call to `weeb` legal (i.e., will the Python interpreter allow it)?

```
weeb("hello")
```

[1 point]

This is legal. For example,

```
def weeb(x):  
    print x  
weeb("hello")
```

In fact, even if the `weeb` function needs the type of `x` to be something different from `str`, the interpreter won't object to the *call*. For example, in the code

```
def weeb(x):  
    print x+1  
weeb("hello")
```

the interpreter doesn't diagnose an error until it encounters the `print` statement.

2. Bob is writing a Python function named `Bleeb`. When `bleeb` is called it should be passed two ints and a float, in this order. `bleeb` returns a float. Write a header (the first line) for `bleeb`. [2 points]

```
def bleeb(int1, int2, float1):
```

The names of the parameters don't matter. However, there must be three parameters.

3. A Python program contains the following assignments.

```
p = 9  
q = 2  
r = 2.0
```

Find the value of each of the following expressions. (For example, the value of `p - q` is 7.) [6]

- (a) `p % q = 1`
- (b) `p/q = 4`
- (c) `p/r = 4.5`
- (d) `p/q*r = 8.0`
- (e) `p > q = True`
- (f) `p > q or r < 1.0 = True`

4. Interpreters and compilers are both programs that translate high-level language programs into machine language. In two sentences or less, what is the main difference between them? [2]

Interpreters translate one statement at a time. After translating a statement, an interpreter executes the translated statement. Compilers translate an entire program, but don't execute any of it: execution of statements comes after compilation.

5. The following program is supposed to read 2 floats, find the square root of their product, and print the result. While the basic design is correct, it contains at least 4 errors that will be detected by the Python interpreter. Find and correct 3 of them. [3]

```
from math import sqrt

# Get rid of ,z
def root_product(x, y, z):
    result = sqrt(x*y)
    return result

first = float(raw_input("Please enter a positive real number\n"))
# Add a close-parenthesis to the end of the statement
second = float(raw_input("Please enter a positive real number\n"))
# Don't indent: "result" should be in line with "second"
    result = root_product(first, second)
# Add a quote before the comma
print "The square root of their product is, result"
```

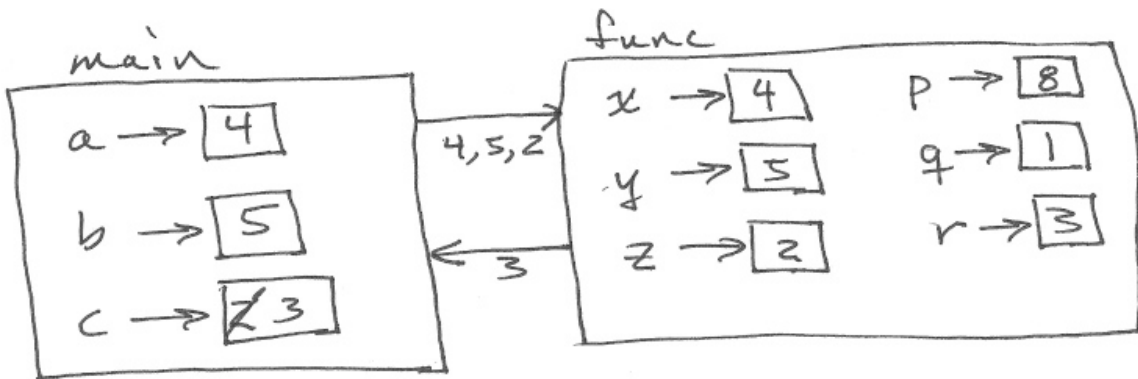


Figure 1: Trace of Program in Problem 6

6. What is the output of the following program when it is run? In other words, what will be printed on the screen? [4]

```
def func(x, y, z):
    print x, y, z
    p = 2*x
    q = y-4
    r = z+1
    print p, q, r
    return r
```

```
a = 4
b = 5
c = 2
print a, b, c
c = func(a, b, c)
print a, b, c
```

Figure 1 shows a trace of the execution of the program. Here's the output:

```
4 5 2
4 5 2
8 1 3
4 5 3
```

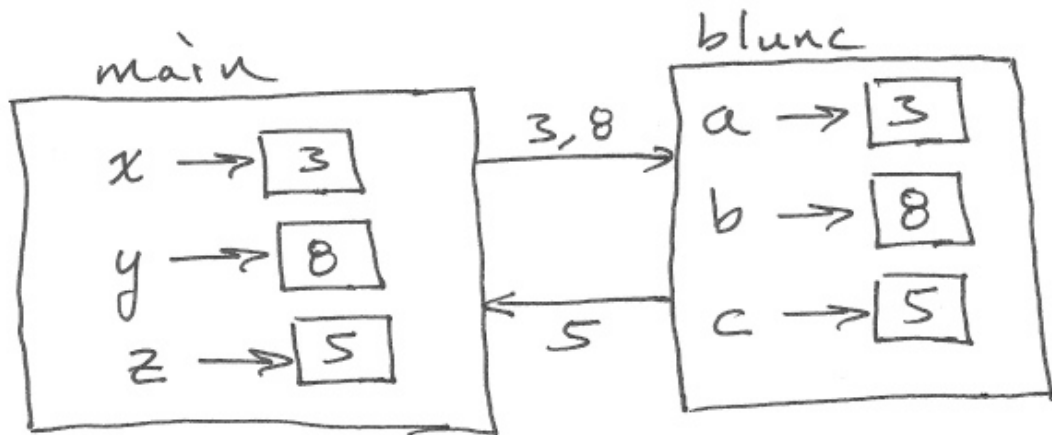


Figure 2: Trace of Program in Problem 7

7. What is the output of the following program when it is run? [3]

```
def blunc(a, b):
    if a >= b:
        c = a - b
    else:
        c = b - a
    return c
```

```
x = 3
y = 8
z = blunc(x, y)
print x, y, z
```

Figure 2 shows a trace of the execution of the program. Here's the output:

3 8 5

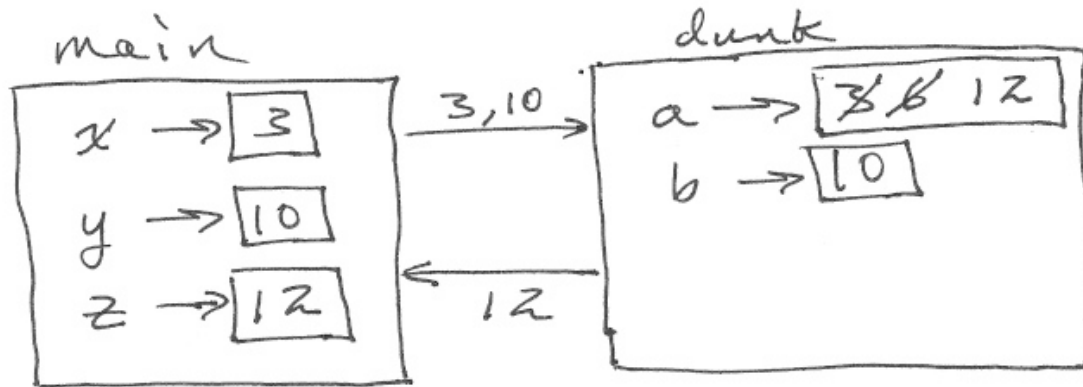


Figure 3: Trace of Program in Problem 8

8. What is the output of the following program when it is run? [4]

```
def dunk(a, b):
    while a < b:
        a = 2*a
    return a
```

```
x = 3
y = 10
z = dunk(x, y)
print x, y, z
```

Figure 3 shows a trace of the execution of the program. Here's the output:

3 10 12

9. Write a Python function called `bigger`. The function is passed two floats. It should print the value of the larger of the two floats (it can print either if they're equal). It should also return the value 1 if the first float is greater than or equal to the second. If the second is bigger than the first, it should return 0. For example, `Bigger(23.5, 16.1)` should print 23.5 and return 1. The function header is

```
def bigger(first, second):
```

Note that the function does *not* read in any data. It is not necessary for you to comment your code. [5]

```
def bigger(first, second):
    if first >= second:
        print first
        return 1
    else:
        print second
        return 0
```

10. Write a Python program that prompts the user to input a series of ints, one per input line. End of input is indicated by an int that's less than or equal to 0. The program finds the product of the positive ints and prints it. If there are no positive ints entered, the program prints the value 1.

For example, if the user enters

```
2
3
5
0
```

then the program will print 30. [5]

```
val = int(raw_input("Enter an int (<= 0 to stop)\n  "))
prod = 1
while val > 0:
    prod = prod*val
    val = int(raw_input("Enter an int (<= 0 to stop)\n  "))
print "The product is", prod
```