JUST LIKE THE REAL TEST – BRING ONLY A WRITING UTENSIL, YOUR ID, AND YOUR BRAIN TO THE TEST.

For the following questions, use these variable definitions.

\[
\begin{align*}
    a &= 4444 \\
    b &= 2 \\
    c &= 11.11 \\
    d &= "hello"
\end{align*}
\]

What is the value and type of each of the following expressions or, if it won't compile, circle that answer

<table>
<thead>
<tr>
<th></th>
<th>type</th>
<th>value</th>
<th>circle if will not compile</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. (2 pts)</td>
<td>(a \div b \div c)</td>
<td></td>
<td>will not compile</td>
</tr>
<tr>
<td>2. (2 pts)</td>
<td>(b + 6 % b)</td>
<td></td>
<td>will not compile</td>
</tr>
<tr>
<td>3. (2 pts)</td>
<td>(a \div d)</td>
<td></td>
<td>will not compile</td>
</tr>
<tr>
<td>4. (2 pts)</td>
<td>(\text{math.ceil}(c) + 1)</td>
<td></td>
<td>will not compile</td>
</tr>
<tr>
<td>5. (2 pts)</td>
<td>(d \times 0)</td>
<td></td>
<td>will not compile</td>
</tr>
<tr>
<td>6. (2 pts)</td>
<td>(\text{str}(d))</td>
<td></td>
<td>will not compile</td>
</tr>
<tr>
<td>7. (2 pts)</td>
<td>(a &gt; b \text{ and } a \leq c)</td>
<td></td>
<td>will not compile</td>
</tr>
<tr>
<td>8. (2 pts)</td>
<td>(b \text{ in } \text{range}(a))</td>
<td></td>
<td>will not compile</td>
</tr>
<tr>
<td>9. (2 pts)</td>
<td>(b \text{ in } \text{range}(-1,13,-2))</td>
<td></td>
<td>will not compile</td>
</tr>
</tbody>
</table>

10. (5pts) The expressions in 8. and 9. are actually incomplete because the for statement is not shown. Circle your answer

True        False
11. (6 pts) What is the output when these statements are executed?

```python
x = 7
while x in range(1, 12, 3):
    if x // 3 != 0:
        print x,
        x -= 4
print x
```

Write the output here

12. (10 pts) Draw arrows to show which values the variables refer to after the statements are executed. Show the values numbers in RAM also. (It doesn't matter where the are in the program.)

1. x = 3  
   Draw the arrow showing where x refers to after line 1.
   x

2. y = 4  
   Draw the arrow showing where y refers after line 2.
   y

3. y += x
   Draw the arrow showing where y refers after line 3.
   y

4. x = y
   Draw the arrow showing where x refers after line 4.
   x

Write the values that will be in RAM

   .
   .
   .
   .
   .
   .
   .
   .
   .
Which of the following is the correct boolean expression to test for a value being in either the range \([23, 77)\) or the range \((-13, -2)\)?

CIRCLE your answer

- a) \(\text{value} > 23 \text{ and value} < 77 \text{ and value} > -13 \text{ and value} < -2\)
- b) \(\text{value} \geq 23 \text{ and value} < 77 \text{ and value} > -13 \text{ and value} < -2\)
- c) \(23 < \text{value} < 77 \text{ and } -13 < \text{value} < -2\)
- d) \(23 < \text{value} < 77 \text{ or } -13 < \text{value} < -2\)
- e) \(-13 \leq \text{value} < -2 < \text{value} \leq 23 < 77\)
- f) \(\text{value} \in [-13, -2] \text{ or } [23, 77]\)
- g) none of the above

True False Circle your one answer

This will compile:

```python
floatVar = float( "one hundred and two point seven" )
```

True False
13. (5 pts)
How many programmers does it take to call a function?
What are their names?
Which can break the subproblem into more subparts? How?
Then how many programmers are involved?

14. (5 pts) We have calculated the type of the quark – it's your job to write code to display it.
We can only test for exactly four kinds of quarks: top quark, bottom quark, up quark and down quark.
Display the type of quark.
The question for this problem is not to write the code – it is to answer this question:
Which problem form is writing the type of quark on the screen?

15. (5 pts) The run time system does what when it first encounters these lines of code (what happens during the compilation phase)?

```python
def meme(prammy, sammy):
    return 2 ** (prammy + sammy)
```
16. (15 pts) There is a built-in function you have never used before named oblatt.

Here is a session showing the help for that function.

```
IDLE 2.7.1
>>> help( oblatt )
Help on built-in function oblatt in module __builtin__:

oblatt(...)
oblatt(object, object) -> float, str

    Return 1st param raised to the power of the 2nd param
    and
    2nd param raised to the power of the 1st param
```

Obviously you cannot define this function because it is built in.
Your job is to complete the following function by writing the needed parameters and python statements that do what the docstring for zeroIfEqualOrSumOfPowers says.

Even though it's not the best, your function MUST use the oblatt built-in function

You do not need to show any testing for your function.
Do NOT write a main function (obviously).
Do NOT write a complete program (obviously).
Write ONLY the definition for zeroIfEqualOrSumOfPowers (don't forget to the parameters!)

```python
def zeroIfEqualOrSumOfPowers(                ):
    ''' returns 0 if both parameters are the same value
    otherwise, returns the larger param raised to the power of the
    smaller plus the smaller param raised to the power or the larger
    '''
```
17. (20 pts) Show that you can break a problem into its parts by writing only the main function.

Write ONLY variable definitions and function calls – we will hire professional python programmers to write the definitions for all the functions you need written. (Of course they have to be able to know what to write based only on your good names for the functions.)

Do NOT write any function defs except main.
Do NOT write a complete program, just define main.
Whenever you need to call a function to solve a subproblem, choose a good name for that function and call it.
We will hire someone to define it for you.
Yes, you should have local variables and use them correctly to solve the problem.

When the other programmers have finished writing the definitions of the functions you called in main, your main function should solve the following problem:

Do these things until the user enters the same age as yours.
- Display sixteen hexagons using the letter Q.
- Ask the user for their age and their middle initial (assume everyone has a middle initial)
- Display pairs of hexagons, the first one of the pair using a character you like and the second of the pair using the user's initial – one of these hexagons pairs for each year they've been alive.

You may NOT write any calls to raw_input in main (even if you need to get only one piece of data, assume that the function you call will do more than just that).

Assume any global constants you need are already defined – just use good names and style for them.

def main( ):
WRITE YOUR ANSWER TO THIS QUESTION IN THE BLUE BOOK

If you need "scratch" paper, use the Blue Book but cross out anything you do not want graded.

rec06 - BLUE BOOK QUESTION

Write your program solution ON PAPER in the lab.
You do not need to write the #! path string or call main at the bottom.
Each function must have a docstring. Be sure to write it BEFORE writing the body of the function.

Revisit the AstroPay program with these additions and enhancements and changes:

Now there's more than one spacecraft – it's a fleet up there, and each has more than one astronaut aboard.
There are no more retirement contributions and there is no more tax.
For each astronaut there is only the pay per whole weeks aloft ($735.22) and the pay for each leftover day ($35.22 per day).
However, NASA has decided to call it a week when there are more than 5 leftover days.
Each astronaut also gets a bonus based on the length of their first and last names combined - $22.22 per letter.
You can use the builtin function len to determine the length of a string.
The commander-in-chief knows how many spacecraft there are in the fleet.
She goes to the computer where your program is running first and enters the number of spacecraft to process.
Curiously the commander-in-chief doesn't know how many astronauts there are on each spacecraft
The commander of each craft knows how many astronauts were aboard and how many whole days the spacecraft was aloft so
the commander of each spacecraft will be doing the input and reporting for her crew. She counts as one of the crew but there's
nothing special about processing her – she just does the processing for however many should be processed.
All the spacecraft in the fleet launch at different times but they all return at the same time and all want to get paid at the same
time – that's what your program does:
You are to write the code to process the arrival of one fleet of spacecraft.

Being in space makes commanders spacey – you'll need to make sure that when they enter the number of days aloft that they
don't input negative numbers or numbers that are larger than the maximum possible safe time aloft: 578 days.
The commander will never get the number of crew wrong, though.
(You can assume they won't type things like three instead of 3, though).
The commander-in-chied will not make a mistake about how many spacecraft there are in the fleet.

Make sure you define (and call) functions for every subproblem.

Go – NO COMPUTERS – WRITE THE PROGRAM ON PAPER
Here is a sample run. Do not solve examples!
User input is shown in **boldface**.
Values that are calculated are shown in *italicized boldface*.
In **$DDD. CC***, DDD is dollars and CC is cents

Welcome Commander-In_Chief
Enter number of spacecraft in the fleet that just landed: **2**

----- Welcome Commander.
How many in your crew? **3**
How many whole days was the crew aloft? **15**

Enter last name of crewmember: **Billings**
Enter first name of crewmember: **Monty**
Monty Billings will be paid **$DDDDD.CC** which includes a bonus of **$DD.CC**

Enter last name of crewmember: **Washington**
Enter first name of crewmember: **Salmon**
Monty Billings will be paid **$DDDDD.CC** which includes a bonus of **$DD.CC**

Enter last name of crewmember: **Questionne**
Enter first name of crewmember: **Cury**
Monty Billings will be paid **$DDDDD.CC** which includes a bonus of **$DD.CC**

----- Welcome Commander.
How many in your crew? **2**
How many whole days was the crew aloft? **233**
Sorry Commander, that must between 1 and 578
How many whole days was the crew aloft? **-2**
Sorry Commander, that must between 1 and 578
How many whole days was the crew aloft? **7**

Enter last name of crewmember: **Billings**
Enter first name of crewmember: **Monty**
Monty Billings will be paid **$DDDDD.CC** which includes a bonus of **$DD.CC**

Enter last name of crewmember: **Billings**
Enter first name of crewmember: **Monty**
Monty Billings will be paid **$DDDDD.CC** which includes a bonus of **$DD.CC**

[---end of program run----]