1. Evaluate the following expressions and indicate the data type of the result

<table>
<thead>
<tr>
<th>Expression</th>
<th>Result</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 - 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.0 - 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 - int(2.0) + float(2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;a&quot; + &quot;b&quot;*2 + &quot;a&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;2.0&quot; + &quot;4.5&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>float(2**2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 // 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 / 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 % 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 - 3 * 3 ** 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>format(5/2, '&gt;7.2f')</td>
<td></td>
<td></td>
</tr>
<tr>
<td>str.lower('PIKACHU!')</td>
<td></td>
<td></td>
</tr>
<tr>
<td>len(str.lower('PIKACHU!'))</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 &gt; 4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 != 4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 &gt; 4 or 3 &lt; 4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 &gt; 4 and 3 &gt; 4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(True and False) or (False and True)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>not (False) or not (False and True)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>'emergency' &gt; 'emily'</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2. Define the following terms:

program:

function:

argument:

algorithm:
3. Analyze the code below and identify any problems or issues that might exist. Circle the issues and briefly describe the error along with a potential solution (i.e. “Syntax error, add a String delimiter”) Rewrite the code so that it functions correctly.

```python
rate = str(input("How much do you make per hour? ‘\n"))
4hours = input("How many hours did you work this week? \n")

if 4hours < 40:
    pay = rate * hours
if 4hours > 40
    pay = rate * 40
    ot_pay = (hours-40) ** (rate*1.5)

print ("Your total pay is, pay + ot_pay")
```
4. Given the following program:

```python
a = 5
b = 6
c = 20
d = 24

if a < b and b * 2 < c:
    print("Python Case 1")
    print("A", 't', "B", 't', "C")

    if a * 2 == c:
        print(a*2, 't', a*2, 't', a*2)
    elif a * 3 == c:
        print(a*3, 't', a*3, 't', a*3)
    elif a * 4 == c:
        print(a*4, 't', a*4, 't', a*4)
    else:
        print('?', 't', '?', 't', '?')
else:
    print("Python Case 2")
    print("a", 't', "b", 't', "c")

    if b * 2 == d:
        print(b*2, 't', b*2, 't', b*2)
    elif b * 3 == d:
        print(b*3, 't', b*3, 't', b*3)
    elif b * 4 == d:
        print(b*4, 't', b*4, 't', b*4)
    else:
        print('?', 't', '?', 't', '?')
```

What will be printed to the screen when the program is run?
5. Write a program that lets the user figure out how many items they can purchase at a local coffee shop. Begin by asking the user to enter in amount of money as a float. Then ask the user to select a product from a pre-determined list. Figure out how many items the user can purchase, noting that the coffee shop does not sell fractional amounts (i.e. you can't buy half a donut)

How much money do you have?: 10.00

What would you like to buy?
Donut   (d) — 1.50
Coffee (c) — 1.00
Bagel   (b) — 2.50
Scone   (s) — 2.75

Enter your choice (d/c/b/s): d

You can purchase 6 donuts with $10.0

Note that you cannot assume that the user will enter a valid product (i.e. they could type in the string “donut” instead of the string “d”). In this case you will need to present the user with some kind of error (i.e. “Sorry, that's not a valid product”) – you do not need to re-prompt them (you can just end the program). Also, you can assume that the user will input valid floating-point numbers when prompted.
6. Write a “calculator” program that asks the user for two numbers as well as an “operation code” ("a" for add, "s" for subtract, "d" for divide or "m" for multiply). Using the information provided perform the specified operation and print the result. Here is a sample running of the program:

```
Number 1: 2.0
Number 2: 3.0
Operation (a/s/d/m): add
Invalid operation! Try again.
Operation (a/s/d/m): a

2.0 + 3.0 = 5.0
```

Note that you cannot assume that the user will enter a valid operation code (i.e. they could type in the string “multiply” instead of the string “m”). In this case you will need to present the user with some kind of error (i.e. “Sorry, that’s not a valid operation code”) and re-prompt them. However, you can assume that the user will input valid floating-point numbers when prompted.

Also note that dividing a number by 0 will result in a runtime error. Prevent this from happening in your program by providing special output in this case (i.e. 5.0 / 0.0 = undefined)