Problem 1: For each of the following items, say whether they are a valid Python expression or a valid Python statement or not valid. (You can assume all variables that are used have already been assigned a value.) For the expressions, if possible, also say what value they refer to. For statements explain what they do. And for invalid items explain why they are invalid.

1. “a b c”
   This is a valid expression. The value is the string a b c.

2. string
   This is a valid expression. It is a variable. The value is the value that this variable is associated with.

3. 7.2 - 1.3
   The is a valid expression. The value is 5.9.

4. print 2 + 3
   This is a valid statement. It will print 5 to the screen.

5. 6 4
   This is not a valid Python statement or expression. It could be made into a valid expression by either putting quotes around the whole thing, which would make it into a string, or by getting rid of the blank between the two numbers, which would make it into an integer.

6. “7 + 2”
   This is a valid expression. The value is the string 7 + 2.

7. x = 3 ÷ 4
   This is a valid statement. It assigns the boolean value False to the variable x.

8. print w1 w2
   This a not a valid statement. The print statement can either have the format

   print expression

   or

   print expression, expression
w1 w2 could either be made into a single expression by putting an operator like + in between w1 and w2 or we could put a comma in between and use the second format of the print statement.

9. return x
   This is a valid statement. It returns the value of x.

10. r = f(3, 5)
    This is a valid statement. It assigns the return value of the function call f(3, 5) to the variable r.

Problem 2: The following piece of code is supposed to print “One” if the variable a equals 1, “Two” if a equals 2, “Three” if a equals 3, and “Other” if a is somethings else. Instead, it prints “One” and “Other” if a equals 1, “Two” and “Other” if a equals 2, “Three” if a equals 3, and “Other” if a is something else. Explain what is going wrong. How can it be fixed?

```python
if a == 1:
    print "One"
if a == 2:
    print "Two"
if a == 3:
    print "Three"
else:
    print "Other"
```

This code snippet consists of three if-statements. The Python interpreter will always check all three if-statements. For example, if the value of a is 1, it will check the first if-statement, find that it’s True and print “One”. It then checks the second if-statement, finds that the condition is false and doesn’t do anything. Then it checks the last if-statement, finds that the condition is false and goes into the else part, where it prints “Other”.

If we replace the second and third if by an elif, then the different cases are interpreted to be alternatives to each other, such that if one condition evaluates to True the elif statements and the else clause are skipped.

Problem 3: Consider the following code snippet. How many times does hallo get printed to the screen when this code is executed?

```python
count = 0
while count < 7:
    count = count + 1
    print "hallo"
```
hallo gets printed seven times. Count starts at 0 and then gets increased by one in each run through the while-loop. The while loop keeps running until count gets to be 7. So, we have seven runs through the while loop (1. count=0, 2. count=1, 3. count=2, ..., 7. count=6) and hallo gets printed in each run.

**Problem 4:** The following code snippet is supposed to sum all integers between 0 and 10. Instead the Python interpreter goes into an infinite loop when this code is run. Explain what is going wrong. How can it be fixed?

```python
total = 0
i = 0
while i <= 10:
    total = total + i
print "The total is: " + str(total)
```

This while loop runs as long as i is smaller or equal to 10. i starts out with the value 0. So the while-loop starts running. However, the value of i then never changes and, therefore, the while loop keeps running indefinitely.

To fix this problem, the value of i needs to change in every run through the while loop. Since the purpose of this codes snippet is supposed to be to add up all numbers up to 10, we increase i by 1 in each run through the loop. That is, add the line i = i + 1 to the body of the loop either before or after the line total = total + i.

**Problem 5:** Define a function that takes a word as its parameter and returns a word that is the input word, but all consonants have been doubled and an ‘o’ has been inserted between them. For example, when called with the parameter value “dog” the function should return “dodogog”, or given the word “elephant” the function should return “elolepophohanontot”. (This is the secret language that Kalle Blomquist and his friends used in the book by Astrid Lindgren, the author of Pippi Longstocking”.)

```python
def kalle_blomquist_code (word):
    encoded_word = ""
    for letter in word:
        if letter=='a' or letter=='e' or letter=='i' \
            or letter=='o' or letter=='u':
            encoded_word = encoded_word + letter
        else:
            encoded_word = encoded_word + letter + 'o' + letter
    return encoded_word
```