1. Python has an inbuilt function called `upper`. It is used to convert a string into uppercase. It works in the following manner:

```
>>> x = 'abc'
>>> x
'abc'
>>> x.upper()
'ABC'
>>> x
'abc'
>>> y = x.upper()
>>> y
'ABC'
```

   Notice that `x.upper()` did not change the value of `x`, but only returned the string converted into uppercase! Also note that we used `x.upper()` instead of `upper(x)`. Now, what do you think the following code should print?

```
x = ['a', 'b']
y = []
for i in x:
    y.append(i)
for i in x:
    y.append(i.upper())
print(y)
```

   Also, can you predict the output of the following piece of code?

```
x = ['a', 'b']
for i in x:
    x.append(i.upper())
print(x)
```

2. Let's now recall `while` loops. Find the output of the following program:

```
i = 0
while i < 3:
    print(i)
    i = i + 1
else:
    print(0)
```

3. Let's write a program that takes as input an integer n and prints the first n *fibonacci* numbers. To recall, first and second fibonacci numbers are 1 and 1 respectively, and for all $n > 2$:

   n^{th} fibonacci number = $(n-1)^{th}$ fibonacci number + $(n-2)^{th}$ fibonacci number

   Fill up the blanks in the following code:

```
def fib(n):
    f = [1,1]
    if n == 1:
        return f[0:1]
    if n == 2:
        return f[__:__]
    for i in range(2,n):
        f.append( _____ + _____ )
    return f

n = ___(_____())
print (fib(n))
```

4. Remember variables have their *scopes*? What should be the output of the program below?

```
c = 1
def func(c):
    for i in (1, 2, 3):
        c = c + i
    return c
print(c)
c = func(c)
print(c)
```