Exercises 2

1. The following code uses a while loop to create a list of the first 10 square numbers:

```
>>> square_numbers = []
>>> x = 1
>>> while x <= 10:
...     square_numbers.append(x ** 2)
...     x += 1

>>> square_numbers
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
```

   a) Re-write the code as a for loop.

   b) Re-write the code as a list comprehension.

   c) Adapt the code that uses the while loop above so that it gives a list of the first 15 square numbers that are even.

2. The code below verifies the following identity for $n = 20$:

$$\sum_{i=0}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$$

```
>>> n = 20
>>> rhs = n * (n +  1) * (2 * n + 1) / 6
>>> lhs = sum(i ** 2 for i in range(n + 1))
>>> lhs == rhs
True
```

   Using a for loop, verify this identity for every interger value of n below 100.

3. In the same way as the previous question, write code to verify the following identity for the first 250 natural numbers:

$$\sum_{i=0}^{n} i^3 = \frac{(n^2+n)^2}{4}$$

4. Write a function for the Heavyside function:

$$H(x) = \begin{cases} 0 & \text{if } x < 0 \\ 0.5 & \text{if } x = 0 \\ 1 & \text{otherwise.} \end{cases}$$

Use this function to evaluate

a) $H(-4)$

b) $H(7)$

c) $H(3.141)$

d) $H(0)$

5. Write a function that give the number of routes of a quadratic equation $ax^2 + bx + c$. It should take in the parameters a, b and c as arguments, and return either 0, 1 or 2 roots.

Use this function to find the number of roots for

a) $x^2 - 3x + 4$

b) $2x^2 - 10x + 1$

c) $4x^2 + 4x + 1$

d) $-7x^2 + 7x - 7$

Note: this is similar to a question on the previous tutorial sheet, but now we need to organise the code as a function.

6. Heron's algorithm for finding the square root of a number A is given by:

Algorithm 1: Heron's algorithm

$\Delta \leftarrow \infty$;
$x \leftarrow A$;
while $\Delta > \epsilon$ **do**
    $\tilde{x} \leftarrow \frac{1}{2}\left(x + \frac{A}{x}\right)$;
    $\Delta \leftarrow |x - \tilde{x}|$;
    $x \leftarrow \tilde{x}$;
end
Output: x

Write a function that implements this algorithm until convergence using a while loop. Choosing a sufficiently small value for ϵ, use this function to find the following values:

a) $\sqrt{7}$

b) $\sqrt{531}$

c) $\sqrt{1000000001}$

d) $\sqrt{60.49371}$

7. Euclid's algorithm for finding the greatest common divisor of two numbers A and B (where $A > B$) is given by:

Algorithm 2: Euclid's algorithm

```
while A > B do
    R ← the remainder when A is divided by B;
    if R = 0 then
        Output: B
        End algorithm.
    else
        A ← B;
        B ← R;
    end
end
Output: x
```

Write a function that implement this algorithm using a while loop. Use this to find the following values:

a) gcd$(1890, 385)$

b) gcd$(2295, 544)$

c) gcd$(136717658, 7043520)$

d) gcd$(32768, 2187)$

8. The Jacobsthal numbers have three equivalent recursive definitions, with base cases $J_0 = 0$ and $J_1 = 1$ given by:

$$J_n = J_{n-1} + 2J_{n-2}$$
$$J_n = 2J_{n-1} + (-1)^{n-1}$$
$$J_n = 2^{n-1} - J_{n-1}$$

Implement all three as recursive Python functions. Then, using a for loop, check that they are all equivalent for the first 30 terms.