THE UNIVERSITY OF THE WEST INDIES
Department of Computing
COMP1126–Introduction to Computing I

Lab 2

1. The roots of a quadratic equation ax^2+bx+c are given by the quadratic formula

$$root1 = \frac{-b+\sqrt{b^2-4ac}}{2a} \qquad root2 = \frac{-b-\sqrt{b^2-4ac}}{2a}$$

Write a function `quadRoots` in python which returns the greater of the two roots. However, the root is returned when the discriminant (i.e. b^2-4ac) is positive when it is negative print a message that there are no real roots.

e.g.

```
quadRoots(15,3,6) -> "No real roots"
quadRoots(6,23,20) -> -1.3333333333333333
```

2. Define a function perfectSquare(n), which returns True if n is a perfect square and False otherwise. Your function takes a positive integer as input and returns a Boolean Value. Use while loops in writing this function. A number n is a perfect square if there is a number from 1 to n-1 whose square is equal to n.

Examples of Perfect square:

- 9 = 3*3
- 16 = 4*4

```
>>> perfectSquare(6)
False
>>> perfectSquare(9)
True
```

3. An integer greater than 1 is said to be prime if it divisible by only 1 and itself. For example 2,3,5,7 are prime numbers, but 4, 6, 8 and 9 are not. Write a function `isPrime` that determines whether a number is prime or not. The function takes a parameter n and checks if there exists a number from 2 to n-1 that n is divisible by [Hint: you can use `for` loops]. If n is divisible by such a number then it is not a prime number and false must be returned. Return true if the number is a prime number. Remember that 1 is not a prime number.

e.g.

```
isPrime(5) -> True
```

4. Use `isPrime` function in a function `primes` that take two numbers as parameters and prints all the prime numbers between those two numbers (e.g. 2 and 10). Ensure that `isPrime` is local function and can only be accessed by function `primes`.

e.g.

```
primes(2,10) -> 2,3,5,7
isPrime(3) -> syntax error
```