```
import math
import check

## Question 1:

def f1(x, y):
    return ((y * 5) ** 2) // x

def f2(a, b):
    return  (a + b) ** (a def % 10)

def f3(n):
    return (math.sqrt(2 * math.pi * n)) * ((n / math.e) ** (n - 1))


## Question 2:

## normal_distribution(x, mean, std_dev) consumes three positive floating
## point numbers, and returns the corresponding value associated with the normal
## distribution with mean mean, and standard deviation std_dev, at x.
## normal_distribution: Float Float Float -> Float
## requires: std_dev, mean, x > 0
## Examples:
## normal_distribution(1.0, 1.0, 1.0) => 0.399
## normal_distribution(12.0, 2.3, 9.0) => 0.028

def normal_distribution(x, mean, std_dev):
    var = std_dev ** 2
    constant_factor = 1 / (std_dev * math.sqrt(2 * math.pi))
```

```
    exponent = ((x - mean) ** 2) / (2 * var)
    function = constant_factor * (1 / math.exp(exponent))
    return function

## Tests:
check.within("Standard Test", normal_distribution(1.0, 1.0, 1.0), 0.399, 0.001)
check.within("Complex Test", normal_distribution(5.332, 3.541, 4.441), 0.083, 0.001)
check.within("Complext Test", normal_distribution(70.321, 70.422, 31.2),0.013 , 0.001)
check.within("Standard Test", normal_distribution(70.0, 65.0, 20.0),0.019 , 0.001)



## Question 3:

## forever_15(n) consumes n, returns 15 always, by following the provided
##     math "trick".
## forever_15: Nat -> Nat  or Nat->15
## requires: n > 0
## Examples:
## forever_15(5) => 15
## forever_15(10000) => 15

def forever_15(n):
    so_far = ((n * 3) + 45) * 2
    return (so_far // 6 ) - n

## Tests:
check.expect("Small number", forever_15(5), 15)
check.expect("Medium number", forever_15(23), 15)
check.expect("Large number", forever_15(342), 15)
```

```
check.expect("Another large number", forever_15(9873420), 15)



## Question 4:


## min3(a, b, c)) returns the maximal value among a b and c without
## using max or min or conditions in Python
## min3: Int Int Int -> Int
## Examples:
## min3(1, 1, 1) => 1
## min3(4, 14, -3) => 14


def min3(a, b, c):
    m2=(a + b - (abs(a - b))) //2
    return (m2 + c - (abs(m2 - c))) //2


## Tests:
check.expect("third is smaller test1", min3(23, 17, 1), 1)
check.expect("third is smaller test2", min3(23, 37, 1), 1)
check.expect("second is smaller test1", min3(17, 1, 5), 1)
check.expect("second is smaller test2", min3(37, 17, 118), 17)
check.expect("third is smaller test1", min3(123, 117, 100), 100)
check.expect("third is smaller test2", min3(223, 337, 100), 100)
check.expect("all the same", min3(5, 5, 5), 5)
check.expect("all negative", min3(-5, -25, -3), -25)
check.expect("mixed", min3(0, -25, 3), -25)
```