R2

1. Given the following code fragment what is the result of the execution: **(1p)**

```
def f(a, b, c):
    a = a + 1
    b.append(3)
    c = c + [3]

a = 7
b = [1, 2]
c = [1, 2]
f(a, b, c)
print(a, b, c)
```

a) print: 7 [1, 2] [1, 2]

b) print: 8 [1, 2, 3] [1, 2, 3]

c) print: 7 [1, 2, 3] [1, 2]

d) error on line: c = c + [3]

2. Please specify and test the following function: **(2p)**

```
def f(l):
    if l == None:
        raise ValueError()
    for e in l:
        if e % 2 == 1:
            return True
    return False
```

3. Asymptotic analysis of the time complexity (best case, average case, worst case). Please also indicate the extra-space complexity. **(2p)**

```
def f(n):
# n – integer number
    s = 0
    m = n
    while m != 0:
        s = s + m / 10
        m = m / 10
    return s
```

```
def a(n):
    s = 0
    for i in range(1, n + 1):
        m = 2 * n + 1
        s = s + f(m)
    return s
```

4. Let us consider a list $\mathbf{a}_1$, $\mathbf{a}_2$, … $\mathbf{a}_n$ of integer numbers. Using the "Divide et Impera" programming method, write, specify and test a function to compute the number of even elements from the list. **(2p)**

5. For the following problem, please indicate the most **APPROPRIATE** programming method *(Backtracking, Divide et Impera, Greedy, Dynamic Programming)* that can be used for solving it. Please justify the method's applicability and analyse the problem solving according to the particularity of the selected programming tehnique *(without implementation)* . **(2p)**

Give all the posibilities to decompose a given natural number **n** as a sum of prime numbers
(**Example:** For **n=15**, the solutions are **3**, **5**, **7** and **2**, **13**)