Programmazione 1 **Esercitazione 1**

Cognome: **Nome:** **Matricola:**

1. Scrivere una procedura che prenda tre numeri come argomenti e restituisca la somma dei quadrati dei due numeri più grandi.

```python
def SommaMinimi(a, b, c):
    if a < b and a < c:
        return b*b + c*c
    if b < a and b < c:
        return a*a + c*c
    return a*a + b*b
```

2. Si considerino le due procedure seguenti:

```python
def Inc(x):
    return x+1
def Dec(x):
    return x-1
```

Si scriva una funzione `Somma(x,y)` che restituisca la somma `x+y`, usando solo chiamate ricorsive, le espressioni condizionali, e le due funzioni `Inc(x)` e `Dec(x)`.

```python
def SommaRec(x, y):
    if y == 0:
        return x
    else:
        return SommaRec(Inc(x), Dec(y))
def Somma(x, y):
    if x < 0 or y < 0:
        print("Entrambi i numeri devono essere positivi!")
    else:
        return SommaRec(x,y)
```

3. Scrivere una o più procedure per trovare la radice quadrata di un numero, utilizzando il metodo di Newton descritto nel notebook `Lab 2.ipynb`.

```python
def UgualeTol(x, y, tol=0.001):  # Uguali a meno di una tolleranza numerica
    return abs(x - y) < tol
def NewtonMigliora(y, x):
    return (y+x/y)/2
def NewtonIter(y, x):
    if UgualeTol(x, y*y):
        return y
    else:
        return NewtonIter(NewtonMigliora(y, x), x)
def RadiceNewton(x):
    return NewtonIter(1, x)
```

4. Il metodo di Newton per trovare la radice cubica di un numero si basa sul fatto che se y è un'approssimazione della radice cubica di x, allora un'approssimazione migliore è data da $\frac{\frac{x}{y^2}+2y}{3}$. Si usi questa formula per implementare una procedura analoga a quella scritta per trovare la radice quadrata.

```
def NewtonCuboMigliora(y, x):
    return (x/(y*y) + 2*y)/3
def NewtonCuboIter(y, x):
    if UgualeTol(x, y*y*y):
        return y
    else:
        return NewtonCuboIter(NewtonCuboMigliora(y, x), x)
def RadiceCubicaNewton(x):
    return NewtonCuboIter(1, x)
```

5. Modificare le procedure `RadiceBruteForce(x,y)`, `RadiceBisection(x)` e la procedura scritta per risolvere l'esercizio 1.2, in modo che vengano contate il numero di chiamate ricorsive. Per ciascuno dei tre metodi, riportare nel riquadro sotto il numero di iterazioni necessarie per trovare la radice quadrata di 13.

```
def CountRadiceBisection(x):
        def CountRadiceBisectionSearch(x, a, b, counter):
                if a > b:
                        return -1
                y = (b + a)/2
                if UgualeTol(x, y*y):
                        return y, counter
                if y*y < x:
                        return CountRadiceBisectionSearch(x, y, b, counter+1)
                else:
                        return CountRadiceBisectionSearch(x, a, y, counter+1)
    return CountRadiceBisectionSearch(x, 1, x, 1)

def CountRadiceNewton(x):
        def CountNewtonIter(y, x, counter):
                if UgualeTol(x, y*y):
                        return y, counter
                else:
                        return CountNewtonIter(NewtonMigliora(y, x), x, counter+1)

        return CountNewtonIter(1, x, 1)
```

6. Il Massimo Comun Divisore (MCD) di due numeri intero a e b è definito come il più grande numero intero che divide sia a che b senza resto. Esiste un metodo famoso, dovuto ad Euclide, per calcolare tale numero. L'idea dell'algoritmo si basa sull'osservazione che, se r è il resto di quando a è diviso per b, allora i divisori comuni di a e b sono esattamente esattamente gli stessi divisori comuni tra b e r. Quindi possiamo usare l'equazione

$$MCD(a, b) = MCD(b, r)$$

per ridurre il problema di trovare i divisori comuni calcolando il MCD tra coppie di numeri interi via via più piccoli. Per esempio:

```
MCD(206, 40) = MCD(40, 6) = MCD(6,4) = MCD(4,2) = MCD(2,0) = 2
```

Scrivere una procedura che calcola il massimo comune divisore usando l'algoritmo di Euclide. **NOTA**: per calcolare il resto di una divisione tra due numeri interi si usa l'operatore modulo `%`, ovvero il simbolo percentuale (esempio: `7%3 = 1`).

```python
def MCD(a, b):
    if b == 0:
        return a
    else:
        return MCD(b, a % b)
```

7. **CHALLENGE (facoltativo)**: Si consideri il problema seguente. Siano date le monetine da 1, 2, 5 e 10 centesimi di euro: quanti modi esistono per cambiare una monetina da 20 centesimi? E se consideriamo anche le monetine da 20 e 50 centesimi, in quanti modi possiamo cambiare una moneta da un euro? Mandare la soluzione per email.