Λύσεις ασκήσεων εργασίας 2

Παύλος Μαργαρίτης

June 13, 2025

Listing 1: **exercise 1**

```
#!/usr/bin/python3

"""
ΑΣΚΗΣΗ 1
Αναθέστε στις μεταβλητές a, b, c, d τις τιμές 3, -121, 1.033 * 10 -33 και
    'Python is fun' και τυπώστε τις τιμές τους στη μορφή:
The value of the vatiable a is 3, κ.ο.κ.
"""

def main():
    a=3
    b=123
    c=1.033 * 10 -33
    d="Python is fun"

    print(f"the value of the variable a is: {a}")
    print(f"the value of the variable b is: {b}")
    print(f"the value of the variable c is: {c}")
    print(f"the value of the variable d is: {d}")

if __name__=="__main__":
    main()
```

Listing 2: **exercise 2**

```python
#!/usr/bin/python3

"""
Οι συναρτήσεις hex, oct, bin μετατρέπουν έναν ακέραιο στο δεκαεξαδικό, οκταδικό και δυαδικό σύστημα, αντίστοιχα.
Εκτυπώστε τον αριθμό μητρώου σας σε κάθε ένα από αυτά τα συστήματα.
Παρατηρήστε πως σημειώνονται οι αριθμοί σε αυτά τα συστήματα.
(Η συνάρτηση int κάνει την αντίστροφη μετατροπή).
"""

import math

def main():
    am=13107
    hex_am=hex(am)
    oct_am=oct(am)
    bin_am=bin(am)

    print(f"Ο αριθμός μητρώου {am} στο δεκαεξαδικό σύστημα είναι: {hex_am}")
    print(f"Ο αριθμός μητρώου {am} στο οκταδικό σύστημα είναι: {oct_am}")
    print(f"Ο αριθμός μητρώου {am} στο δυαδικό σύστημα είναι: {bin_am}")

    print("\nΠαρατηρήσεις:")
    print("Οι δεκαεξαδικοί αριθμοί ξεκινούν με το πρόθεμα '0x'. ")
    print("Οι οκταδικοί αριθμοί ξεκινούν με το πρόθεμα '0o'. ")
    print("Οι δυαδικοί αριθμοί ξεκινούν με το πρόθεμα '0b'.")

if __name__=="__main__":
    main()
```

Listing 3: **exercise 3**

```
#!/usr/bin/python3

"""
ΑΣΚΗΣΗ 3
Να γράψετε πρόγραμμα σε πυτηον που χρησιμοποιεί συνάρτηση main και θα δέχεται ως είσοδο με έλεγχο τιμών το ύψος ενός ατόμου και θα τυπώνει στην οθόνη το παρακάτω:
    Δώσε ύψος:
    Το ύψος του ατόμου είναι: 1.83 μέτρα
    Είναι ψηλός
Για τον χαρακτηρισμό του ύψους να χρησιμοποιηθούν οι εκφράσεις: (πολύ κοντός, κοντός, κανονικός, ψηλός, πολύ ψηλός) με αντίστοιχα ύψη 1.40, 1.72, 1.80, 1.95,και 2.05.
"""

def main():
    height=None
    while True:
        try:
            height=float(input("Δώσε ύψος: "))

            if (height<1.40 or height>2.05):
                print("Το ύψος πρέπει να είναι μεταξύ 1.40 και 2.05 μέτρων.")
                continue
            break

        except Exception as e:
            print(f"Exception: {str(e)}")

    print(f"Το ύψος του ατόμου είναι {height:.2f} μέτρα")
    """
    Έτσι εκτυπώνω μόνο δύο δεκαδικά ψηφία του height.
    """

    if height<=1.40:
        print("Είναι πολύ κοντός")

```

```
34      elif height<=1.72:
35          print("Είναι κοντός")
36
37      elif height<=1.80:
38          print("Είναι κανονικός")
39
40      elif height<=1.95:
41          print("Είναι ψηλός")
42
43      else:
44          print("Είναι πολύ ψηλός")
45
46
47
48  if __name__=="__main__":
49      main()
```

Listing 4: **exercise 4**

```
#!/usr/bin/python3

"""
ΑΣΚΗΣΗ 4
Στην κλίμακα Fahrenheit το νερό παγώνει στους 32 βαθμούς (0 oC) και βράζει στους 212 (100 oC)
Να γραφεί συνάρτηση μετατροπής από oC σε oF και το αντίστροφο και πρόγραμμα (main) που ο χρήστης να δίνει oC ή oΦ  και να επιστρέφει την αντίστοιχη θερμικρασία στην άλλη κλίμακα.
"""

def F2C(F_deg):
    # Μετατροπή F σε C, χρησιμοποιώντας τον τύπο: C = 5/9 * (F - 32)
    C_deg=( 5 / 9 ) * (F_deg - 32)
    return C_deg

def C2F(C_deg):
    # Μετατροπή C σε F, χρησιμοποιώντας τον τύπο: F = 9/5 * C +32
    F_deg=( 9 / 5 ) * C_deg + 32
    return F_deg

def main():
    C=None
    F=None
    opt=None
    try:
        # block για Έλεγχο στην είσοδο τιμών
        while True:
            print("Choose an option\n \t convert oF to oC (press F2C) \n \t convert oC to oF (press C2F)")
            opt=input("please give your answer: ")
            if not(opt=="C2F" or opt=="F2C"):
                print("error. Invalid option")
                continue
            break
        # Τέλος block

    except Exception as e:
```

```
        print(f"Exception {str(e)}")
        exit(1)

    try:
        if (opt=="C2F"):
            while True:
                C=float(input("Give temperature in oC: "))
                if (C< -273.15):
                    print("This temperature does not exist. Try again.")
                    continue
                else:
                    break

            print(f"{C:.2f} oC = {C2F(C):.2f} oF")

        if (opt=="F2C"):
            while True:
                F=float(input("Give temperature in oF: "))
                if (F< -459.67):
                    print("This temperature does not exist. Try again.")
                    continue
                else:
                    break

            print(f"{F:.2f} oF = {F2C(F):.2f} oC")

    except Exception as e:
        print(f"Exception: {str(a)}")
        exit(1)

if __name__=="__main__":
    main()
```

Listing 5: **exercise 5**

```python
#!/usr/bin/python3

"""
ΑΣΚΗΣΗ 5
Να βρείτε το εμβαδόν του χωρίου που περικλύεται μεταξύ του εγγεγρμένου και του περιγεραμμένου κύκλου ενός τετραγώνου πλευράς 3.
"""

import math
import random

# Συνάρτηση που επισρτέφει το εμβαδόν τετραγώνου πλευράς a (προεπιλογή a=3)
def sq_area(a=3):
    return pow(a,2)

# Συνάρτηση που επιστρέφει το εμβαδόν κύκλου ακτίνας r
def circle_area(r):
    return math.pi * pow(r,2)

# Συνάρτηση που επιστρέφει το εμβαδόν του εγγεγραμμένου κύκλου τετραγώνου πλευράς a (προεπιλογή a=3)
def c_area_1(a=3):
    "Η ακτίνα του εγγεγραμμένου κύκλου είναι η μισή πλευρά του τετραγώνου"
    return math.pi * pow(a/2,2)

# Συνάρηση που επιστρέφει το εμβαδόν περιγγεγραμμένου κύκλου τετραγώνου πλευράς a (προεπιλογή a=3)
def c_area_2(a=3):
    "Η ακτίνα του περιγεγραμμένου κύκλου είναι η μισή της διαγωίου του τετραγώνου"
    "Η διαγώνιος του τετραγώνου πλευράς a είναι sqrt(2)*a"
    return math.pi * pow( (a * math.sqrt(2)) / 2 , 2 )

def main():
    area=circle_area(1.5 * math.sqrt(2)) - circle_area(1.5)
    print("Το εμβαδόν του χωρίου που περικλύεται μεταξύ του εγγεγραμμένου και του περιγεγραμμένου κύκλου \nενός τετραγώνου πλευράς 3 είναι:
    ",area, "τ.μ.")
```

```
33      print("Επαληθεύουμε το αποτέλεσμα και με τις συναρτήσεις c_area_1 και
            c_area_2")
34      area=c_area_2() - c_area_1()
35      print("Το εμβαδόν του χωρίου που περικλύεται μεταξύ του εγγεγραμμένου κα
            ι του περιγεγραμμένου κύκλου \nενός τετραγώνου πλευράς 3 είναι:
            ",area, "τ.μ.")
36
37
38      print("Πιο γενικά και τυχαία:")
39      a=random.randrange(1,100)
40      print(f"Επιλέγουμε τυχαίο τετράγωνο πλευράς {a}")
41      area=c_area_2(a)-c_area_1(a)
42      print(f"Το εμβαδόν μεταξύ του εγγεγραμμένου και του περιγεγραμμένου κύκλ
            ου τετραγώνου πλευράς {a} είναι: {area} τ.μ.")
43
44  if __name__=="__main__":
45      main()
```

Listing 6: **exercise 6**

```python
#!/usr/bin/python3

"""
ΑΣΚΗΣΗ 6
Ένα έτος είναι δίσεκτο αν διαιρείται με το 4 αλλά όχι με το 100, με εξέραιση τα έτη που διαιρούνται με το 400, τα οποία είναι δίσεκτα.
Γράψτε ένα πρόγραμμα που να καλεί συνάρτηση σε Python η οποία να ελέγχει αν ένα έτος είναι δίσεκτο ή όχι.
"""

import random

# Συνάρτηση που επιστρέφει True ή False ανάλογα με το αν το έτος year είναι δίσεκτο ή όχι.
def checkLeapYear(year):
    if year%4==0:
        if year%100==0:
            if year%400==0:
                return True
            else:
                return False
        else:
            return False
    else:
        return False

def main():
    # Δημιουργία λίστας με αρχικά 4 προκαθορισμένα έτη και υπόλοιπα 15 τυχαία έτη
    years=[4,100,400,123]+[random.randint(0,2400) for _ in range (15) ]
    print("Τα έτη που επιλέγουμε να ελέγξουμε είναι: ")
    print(*years,sep=',')

    for el in years:
        print(f"Είναι το έτος {el:>5} δίσεκτο; {checkLeapYear(el)}")


if __name__=="__main__":
```

```
    main()

"""
ΣΤΟΙΧΙΣΗ ΚΕΙΜΕΝΟΥ ΕΞΟΔΟΥ
.ljust(width, fillchar=' ') (left Justify):
στοιχίζει την συμβολοσειρά αριστερά σε ένα πεδίο καθορισμένου width. Αν η συμβολοσειρά είναι μικρότερη από το width τότε προστίθενται χαρακτήρες fillchar (προεπιλογή είναι το ""), ώστε να συμπληρωθεί το πλάτος

.rjust(width, fillchar=' ') (right Justify)

.center(width, fillchar=' ') (Center)

Μπορούμε να ενσωματώσουμε αυτές τις μεθόδους απευθείας μέσα σε f-string. Η γενική σύνταξη είναι:
f"{value:< width}" για αριστερή στοίχιση
f"{value:> width}" για δεξιά στοίχιση
f"{value:^ width}" για κεντρική στοίχιση
Πιο συγκεκριμένα:
f"{value:fillcharWhithout"" < width}"
f"{value:fillcharWhithout"" > width}"
f"{value:fillcharWhithout"" ^ width}"
"""
```

Listing 7: **exercise 7**

```
#!/usr/bin/python3

"""
ΑΣΚΗΣΗ 7
Η ταχύτητα του φωτός είναι 3000 000 km/sec.
Ένα έτος φωτός είναι η απόσταση που διανύει το φως σε ένα έτος, δηλαδή περίπου 9.5 τρισεκατομμύρια km.
Πόσο χρόνο χρειάζεται το αυτοκίνητό σας, αν ταξιδεύει με σταθερή ταχύτητα 120 km/h, για να διανύσει ένα δευτερόλεπτο φωτός;
Να υλοποιήσετε κατάλληλη συνάρτηση που να δέχεται ως είσοδο ταχύτητα σε km/h και απόσταση σε km και να υπολογίζει δευτερόλεπτα φωτός
"""

# ταχύτητα u=s/t άρα t=s/u
# Ένα δευτερόλεπτο φωτός είναι η απόσταση που διανύει το φως (θ=3000000 Km/sec) σε t=1sec, δηλαδή s=300 000km

def TimeInLightSeconds(speed_kmh,distance_km):
    """
    Υπολογίζει τον χρόνο σε sec που χρειάζεται ένα όχημα που έχει ταχύτητα speed_km/h για να διανύσει μια απόσταση distance_km
    """
    if speed_kmh<=0:
        return "Η ταχύτητα πρέπει να είναι θετική."

    time_seconds = distance_km / (speed_kmh / 3600) # Μετατροπή km/s σε km/sec
    return time_seconds

def main():
    c=300000 #km/sec
    car_speed = 120 # km/h
    # Απόσταση ενός δευτερολέπτου φωτός σε km
    light_second_distance = c*1
    time_to_travel_light_second = TimeInLightSeconds(car_speed,light_second_distance)

    print(f"Η ταχύτητα του αυτοκινήτου είναι: {car_speed} km/h")
```

```
32      print(f"Η απόσταση ενός δευτερολέπτου φωτός είναι:
           {light_second_distance} km")
33      print(f"Ο χρόνος που χρειάζεται το αυτοκίνητο για να διανύσει απόσταση
           = (1 δευτερόλεπτο φωτός) είναι: {time_to_travel_light_second:.2f}
           sec")
34
35  if __name__=="__main__":
36      main()
```

Listing 8: **exercise 8**

```
#!/usr/bin/python3

"""
ΑΣΚΗΣΗ 8
Να γεμίσετε έναν πίνακα 3x3 με τυχαίους ακεραίους στο [0,1].
Να χρησιμοποιήσετε για το σκοπό αυτό κατάλληλες συναρτήσεις.
Να μετασχηματίσετε τον πίνακα από 3x3 σε 4x6 και εν συνεχεία σε 6x6 μέσω κατ
    άλληλψν μετασχηματισμών.
"""

import random
import math

matrix=None

def generate_matrix():
    return [[random.randint(0,1) for _ in range(3)] for _ in range(3)]

def add_cols(*cols):
    global matrix
    help_matrix = matrix
    num_rows = len(matrix)
    if len(matrix) == 0: #Ισοδυναμεί με το: if not matrix γιατί
       bool([])==False
        return
    for i,el in enumerate(cols):
        if len(el) != num_rows:
            print(f"invalid dimension: length of matrix (={num_rows}) !=
               length of col {i+1} (={len(el)}")
            matrix = help_matrix
            return
    for el in cols:
        for j in range(num_rows):
            matrix[j].append(el[j])
    return

def add_rows(*rows):
    global matrix
```

```
36      help_matrix=matrix
37      if len(matrix) == 0:
38          return
39
40      if matrix[0]:
41          num_cols = len(matrix[0])
42      else:
43          num_cols = 0
44
45      for i,el in enumerate(rows):
46          if len(el) != num_cols:
47              print(f"invaqlid dimension: length of matrix row (={num_cols})
                    != length of row {i+1} (={len(el)}):")
48              matrix = help_matrix
49              return
50          matrix.append(list(el))
51      return
52
53  def print_matrix(matrix):
54      """Εκτυπώνει έναν πίνακα με στοίχιση"""
55      if len(matrix) == 0:
56          return
57      """ΠΡΟΣΟΧΗ"""
58      #Βρίσκουμε το max πλάτος που χρειάζεται κάθε στοιχείο του πίνακα για να
            χωρέσει όταν μετατραπεί σε string
59      max_width = max( len(str(el)) for row in matrix for el in row ) + 1
60
61      #print("-"*(len(matrix[0])*max|_width if matrix[0] else 0)
62
63      for row in matrix:
64          print(" ".join(str(el).rjust(max_width) for el in row))
65
66  def main():
67      global matrix
68      matrix = generate_matrix()
69      print("Ο αρχικός 3x3 πίνακας είναι:")
70      print_matrix(matrix)
71
```

```
    add_rows([5,5,5])
    add_cols([4,4,4,4],[4,4,4,4],[4,4,4,4])

    print("\n0 νέος 4x6 πίνακας είναι:")
    print_matrix(matrix)

    add_rows([8,8,8,8,8,8],[8,8,8,8,8,888])

    print("\n0 νέος 6x6 πίνακας είναι:")
    print_matrix(matrix)


if __name__=="__main__":
    main()
```

Listing 9: **exercise 9**

```
#!/usr/bin/python3


"""
ΑΣΚΗΣΗ 9
Να γραφεί πίνακας nxn στοιχείων και να τον γεμίσετε σε ξεχωριστή συνάρτηση με ψευδοτυχαίες ακέραιες τιμές στο διάστημα [1,6].
Εν συνεχεία, σε ξεχωριστή συνάρτηση να εμφανίζεται τον πίνακα καθως και σε ξεχωριστή συνάρτηση να υπολογίζεται την συχνότητα εμφάνισης κάθε αριθμού (1,2,3,4,5,6) και να εμφανίζεται το αποτέλεσμα.
"""

import random

def generate_matrix(num_rows,num_cols):
    return [ [random.randint(1,6) for _ in range(num_rows) ] for _ in range(num_cols) ]


def print_matrix(matrix):
    """Εκτυπώνει έναν πίνακα με μορφοποίηση"""
    print("Ο πίνακας είναι")
    if len(matrix)==0:
        return

    max_str_width=max( len(str(el)) for row in matrix for el in row) +2

    for row in matrix:
        print( "".join(str(el).rjust(max_str_width) for el in row) )


def frequency(matrix):
    """Αρχικοποίηση ενός άδειου λεξικού: τιμή στοιχείου - συχνότητα στοιχείου"""
    freq_dict={i:0 for i in range(1,7)}

    for row in matrix:
        for el in row:
```

```
            freq_dict[el]+=1

    for key,value in freq_dict.items():
        print(f"the frequency of {key} is {value}")

def main():
    n=None
    m=None
    try:
        while True:
            n=int(input("Δώσε το μέγεθος του πίνακα: "))
            if n<=0:
                print("invalid size")
                continue
            break
    except Exception as e:
        print(f"Exception: {str(e)}")
        exit(1)

    matrix=generate_matrix(n,n)
    print_matrix(matrix)
    frequency(matrix)


if __name__=="__main__":
    main()
```

Listing 10: **exercise 10**

```
#!/usr/bin/python3


"""
ΑΣΚΗΣΗ 10
Να γραφεί πρόγραμμα που θα χρησιμοποιεί συνάρτηση για να γεμίζει k διανύσματα n στοιχείων (k<n) με ψευδοτυχαίες πραγματικές τιμές στο [1,100] και εν συνεχεία συνάρτηση που θα τυπώνει τυα διανύσματα αυτά στην οθόνη καθώς και συνάρτηση που θα υπολογίζει και θα τυπώνει τα μέτρα των διανυσμάτων αυτών στον Rn
"""

import random
import math
from math import hypot

def generate_vectors(dimension,count=1):
    if count>dimension:
        print("count>dimension".center(25,'*').upper())
        return
    return [ [round(random.uniform(1,100),2) for _ in range(dimension)] for _ in range(count) ]

def print_vec(vec_list):
    for i,el in enumerate(vec_list):
        print(f"Το διάνυσμα {i+1:3d} είναι: {el}")


def norm_vec(vec_list):
    print("Υπολογίζουμε τα μέτρα των διανυσμάτων στον Rn")

    norm_dict={}
    for el in vec_list:
        norm_dict[tuple(el)]=math.sqrt(sum(x**2 for x in el))
    # ΠΡΟΣΟΧΗ: ως κλειδιά λεξικού μπορούν να χρησιμοποιηθούν μόνο immutable μεταβλητές
    # Για παράδειγμα, θα ήταν λάθος να γράψουμε
    """
```

```
33          norm_dict[el]=math.sqrt(sum(x**2 for x in el))
34      """
35      # γιατί το el είναι λίστα (mutable)
36      # Επομένως βάζοντας το στοιχείο σε 1X1 tuple, το κάνουμε immutable
37
38      """
39      ΕΝΑΛΛΑΚΤΙΚΑ από την Python3.8 μπορούμε να χρησιμοποιήσουμε την hypot() γ
            ια περισσότερα από δύο ορίσματα  για τον υπολογισμό της νόρμας
40
41      norm_dict={tuple(el):math.hypot(*el) for el in vec_list}
42      """
43      for key,value in norm_dict.items():
44          print(f"Το μέτρο του διανύσματος {list(key)} είναι: {value:.2f}")
45      return
46
47  def main():
48      k=None
49      n=None
50      try:
51          while True:
52              try:
53                  k=int(input("Δώσε το πλήθος των διανυσμάτων, δηλαδή το k:
                        "))
54                  if k<=0:
55                      print("Μη έγκυρο πλήθος k. Το k πρέπει να είναι θετικός
                            ακέραιος.")
56                      continue
57                  break
58              except ValueError:
59                  print("Μη έγκυρη εισαγωγή")
60
61          while True:
62              try:
63                  n = int(input("Δώσε τη διάσταση των διανυσμάτων, δηλαδή το
                        n: "))
64                  if n<=0 or n<=k:
65                      print("Μη έγκυρη διάσταση n. Το n πρέπει να είναι θετικό
                            ς αριθμός και μεγαλύτερος από το πλήθος των διανυσμάτ
```

```
                        ων k.")
66                      continue
67                  break
68              except ValueError:
69                  print("Μη έγκυρη εισαγωγή")
70
71      except Exception as e:
72          print(f"Exception: {str(e)}")
73          exit(1)
74
75      vec_list=generate_vectors(n,k)
76      print_vec(vec_list)
77      norm_vec(vec_list)
78
79  if __name__=="__main__":
80      main()
```

Listing 11: **exercise 11**

```
#!/usr/bin/python3

"""
ΑΣΚΗΣΗ 11
Να γραφεί συνάρτηση η οποία δεδομένης μιας θερμοκρασίας T να τυπώνει το μήνυ
    μα
Cold αν η θερμοκρασία είναι μικρότερη από 10 βαθμούς,
Hot αν η θερμοκρασία είναι μεγαλύτερη από 35 βαθμούς,
Pleasant διαφορετικά.
Γράψτε ένα πρόγραμμα Python το οποίο διαβάζει έναν πραγματικό αριθμό x και υ
    πολογίζει την τιμή της συνάρτησης
"""

def weather(T):
    if T<10:
        print("Cold")
    elif 10<=T<=35:
        print("Pleasant")
    else:
        print("Hot")
    return

def main():
    T= None
    try:
        T=float(input("Give temperature T: "))
    except Exception as e:
        print(f"Exception: {str(e)}")
        exit(1)

    weather(T)

if __name__=="__main__":
    main()
```

Listing 12: **exercise 12**

```
#!/usr/bin/python3

"""
ΑΣΚΗΣΗ 12
Γράψτε ένα πρόγραμμα Python που διαβάζει έναν πραγματικό αριθμό x και υπολογίζει την τιμή της συνάρτησης f:
3*x**2+1 if x<=0
2*x+1 if 0<x<=3
sqrt(46+x) else
"""

import math

def f(x):
    if x<=0:
        return 3*x**2+1
    if 0<x<=3 :
        return 2*x+1
    else:
        return math.sqrt(46+x)

def main():
    x=None
    try:
        x=float(input("Give x: "))
    except Exception as e:
        print(f"Exception: {str(e)}")
        exit(1)

    print(f"f({x}) = {f(x)}")

if __name__=="__main__":
    main()
```

Listing 13: **exercise 13**

```
#!/usr/bin/python3

"""
ΑΣΚΗΣΗ 13
Να γράψετε συνάρτηση που να υπολογίζει το μεγαλύτερο περιττό αρθιθμό μεταξύ των ακεραίων x, y και z
"""

# Βοηθητική μεταβλητή για να παίρνουμε απευθείας το όρισμα από την συνάρτηση findMaxOdd()
given_numbers=None

def findMaxOdd(*integers):

    global given_numbers
    given_numbers = integers

    if len(integers)==0:
        return
    else:
        maxOdd=None
        for el in set(integers):
            if el%2==1:
                if maxOdd is None or el>maxOdd:
                    maxOdd=el
        return maxOdd

"""
ΠΡΟΣΟΧΗ:
Η σειρά συγκρίσεων μέσα σε μια εντολή if (ειδικά όταν χρησιμοποιούνται λογικοί τελεστές όπως or, and) είναι καθοριστική για την εκτέλεση του κώδικα.
Όταν χρησιμοποιείται ο τελεστής or, αν η πρώτη συνθήκη είναι True, τότε οι υπόλοιπες συνθήκες δεν αξιολογούνται και η συνολική συνθήκη είναι True.
Όταν χρησιμοποιείται ο τελεστής and , αν η πρώτη συνθήκη είναι False, τότε οι υπόλοιπες συνθήκες δεν αξιολογούνται και η συνιλική συνθήκη είναι False.
"""

```

```
33 def main():
34     m = findMaxOdd(5,7,8)
35     print(f"Ο μεγαλύτερος περιττός αριθμός από τους {given_numbers} είναι
           {m}")
36
37     m = findMaxOdd(5,7,8,9,10,11,12)
38     print(f"Ο μεγαλύτερος περιττός αριθμός από τους {given_numbers} είναι
           {m}")
39
40     m = findMaxOdd(4,6,8)
41     print(f"Ο μεγαλύτερος περιττός αριθμός από τους {given_numbers} είναι
           {m}")
42
43 if __name__=="__main__":
44     main()
```

Listing 14: **exercise 14**

```
#!/usr/bin/python3

"""
ΑΣΚΗΣΗ 14
Γράψτε πρόγραμμα το οποίο ζητάει μια γωνία θ στο διάστημα (0,π/2) και συνάρτ
   ηση που τυπώνει τους sinθ, cosθ, tanθ στη μορφή:
theta = 0.5235987755982988
sin(theta) = 0.8660254037844386
cos(theta) = 0.5000000000000000
tan(theta) = 1.7320508075688767
"""

import math

def trig(theta):
    print(f"theta = {theta:.16f}")
    print(f"sin(theta) = {math.sin(theta)}",f"cos(theta) =
       {math.cos(theta)}",f"tan(theta) = {math.tan(theta)}", sep="\n")

def main():
    theta=None
    try:
        while True:
            theta=float(input("Give theta in (0,π/2) : "))
            if not(0<theta<math.pi/2):
                print("theta must be in (0,π/2)")
                continue
            break
    except Exception as e:
        print(f"Exception: {str(e)}")
        exit(1)

    trig(theta) #:.16f είναι η προεπιλογή

if __name__=="__main__":
    main()
```

Listing 15: **exercise 15**

```
#!/usr/bin/python3

"""
ΑΣΚΗΣΗ 15
Γράψτε κατάλληλη συνάρτηση για την επεξεργασία της ακολουθίας χαρακτήρων
    AppliedMathematics.
Συγκεκριμένα, γράψτε εντολές οι οποίες τυπώνουν:
(α) τους πρώτους 6 χαρακτήρες
(β) τους τελευταίους 5 χαρακτήρες
(γ) Κάθε δεύτερο χαρακτήρα ξεκινώντας από την θέση 3
(δ) όλους τους χαρακτήρες ξεκινώντας από το 2 από το τέλος χαρακτήρα και προ
    χωρώντας προς την αρχή
(ε) μια ακολουθία αποτελούμενη από τους χαρακτήρες στις άρτιιες θέσεις ακολο
    υθούμενη από τους χαρακτήρες στις περιττές θέσεις
Να γενικευτεί η παραπάνω συνάρτηση για οποιοδήποτε αλφαριθμητικό 18 χαρακτήρ
    ων
"""

def edit(mstr):
    print(mstr[0:6])
    print(mstr[-5:])
    print(mstr[3::2])
    print(mstr[-2::-1])
    print(mstr[::2]+mstr[1::2])

def main():
    edit("AppliedMathematics")

if __name__=="__main__":
    main()
```

Listing 16: **exercise 16**

```
#!/usr/bin/python3

"""
ΑΣΚΗΣΗ 16
Γράψτε συνάρτηση που ζητά από το χρήστη να δώσει ένα αλφαριθμητικό και ελέγχει ποια φωνήεντα έχει (επιστρέφει τον πίνακα των φωνηέντων) (a, e, i, o, u)
Το κυρίως πρόγραμμα εμφανίζει τον πίνακα αυτόν.
Να γενικευτεί η συνάρτηση και για τα φωνήεντα της ελληνικής γλώσσας
"""

# Αποθηκεύουμε τα φωνήεντα σε σύνολα (sets)
eng=eng_vowels={'a','e','i','o','u','A','E','I','O','U'}
gr=gr_vowels={'α','ε','ι','υ','η','ο','ω','Α','Ε','Ι','Η','Υ','Ο','Ω'}
all=vowels=eng_vowels|gr_vowels # | is for union sets

def vowels(language=all):
    input_str=None
    try:
        input_str = input("Give a string: ")
    except Exception as e:
        print(f"exception: {str(e)}")
        exit(1)

    vows=set() #initialize an empty set
    for ch in input_str:
        if ch in language:
            vows.add(ch)
    return vows

def main():
    V=vowels(eng)
    print(f"Τα αγγλικά φωνήεντα που βρέθηκαν είναι: {V}")

    V=vowels(gr)
    print(f"Τα ελληνικά φωνήεντα που βρέθηκαν είναι: {V}")

    V=vowels()
```

```
    print(f"Τα αγγλικά και ελληνικά φωνήεντα που βρέθηκαν είναι: {V}")

if __name__=="__main__":
    main()
```