**Ονοματεπώνυμο:** **Ημερομηνία:**

## Γραπτή Αξιολόγηση – Αναζήτηση & Ταξινόμηση (Python 2)

Διάρκεια: 30 λεπτά – Σύνολο μονάδων: 100

### ΜΕΡΟΣ Α – Σωστό / Λάθος (20 μονάδες – 4 μονάδες η καθεμία)

**1. Στη σειριακή αναζήτηση, ο έλεγχος γίνεται με όλα τα στοιχεία της λίστας.**

**2. Η δυαδική αναζήτηση μπορεί να εφαρμοστεί σε οποιαδήποτε λίστα, ταξινομημένη ή όχι.**

**3. Στον αλγόριθμο ταξινόμησης σε αύξουσα σειρά, γίνεται ανταλλαγή όταν το στοιχείο στα δεξιά είναι μικρότερο από το στοιχείο στα αριστερά.**

**4. Η ταξινόμηση σε φθίνουσα σειρά χρησιμοποιεί την ίδια δομή επαναλήψεων με την αύξουσα· μόνο η συνθήκη σύγκρισης αλλάζει.**

**5. Στις παράλληλες λίστες (grades, names), σε περίπτωση ίσων βαθμών τα ονόματα ταξινομούνται αλφαβητικά από το Α προς το Ζ.**

### ΜΕΡΟΣ Β – Πολλαπλής Επιλογής (40 μονάδες – 8 μονάδες η καθεμία)

**6. Στη δυαδική αναζήτηση, ο δείκτης middle υπολογίζεται ως:**
(A) (first + last)/2 (B) (first + last)*2 (C) last/2 (D) first+1

**7. Στη σειριακή αναζήτηση, ποια μεταβλητή χρησιμοποιείται για να δείξει ότι το στοιχείο δεν βρέθηκε;**
(A) middle (B) found (C) thesi (D) index

**8. Στον αλγόριθμο ταξινόμησης μιας λίστας numbers σε αύξουσα σειρά, η ανταλλαγή των θέσεων δύο στοιχείων εκτελείται όταν:**
(A) numbers[j] > numbers[j-1] (B) numbers[j] < numbers[j-1] (C) numbers[i] < numbers[j] (D) numbers[j] == numbers[j-1]

**9. Στην ταξινόμηση παράλληλων λιστών:**
(A) πρώτα ταξινομείται η πρώτη λίστα και μετά η δεύτερη (B) η πρώτη λίστα ταξινομείται σε αύξουσα σειρά και η δεύτερη σε φθίνουσα
(C) μία αντιμετάθεση γίνεται και στις δύο λίστες (D) οι λίστες μπορούν να έχουν διαφορετικό μέγεθος

**10. Στη δυαδική αναζήτηση, όταν το στοιχείο που ψάχνουμε x είναι μικρότερο από το numbers[middle], μετακινούμε:**
(A) το first δεξιά του middle (B) το last αριστερά του middle (C) σταματάμε (D) middle=0

## ΜΕΡΟΣ Γ – Συμπλήρωση Κενών σε Κώδικα (40 μονάδες – 10 μονάδες η καθεμία)

**11. Συμπλήρωση κώδικα Σειριακής Αναζήτησης:**

```
for i in _____ :     # (A) range(len(numbers))   (B) numbers   (C) len(numbers)
  if numbers[i] _____ x :     # (A) ==   (B) <   (C) >
    thesi = _____         # (A) i   (B) x   (C) -1
```

**12. Συμπλήρωση κώδικα Δυαδικής Αναζήτησης (10 μονάδες):**

```
numbers = [4, 8, 15, 16, 23, 42, 44, 49, 52, 55, 59]
x = input()

first = 0
last  = len(numbers) - 1
found = _____       # (A) True   (B) False   (C) 0

while first <= last and found == False:
  middle = _____   # (A) (first + last)/2   (B) (last - first)/2   (C) last/2

  if numbers[middle] == x:
    found = True
    thesi = _____    # (A) middle   (B) first   (C) last
  else:
    if x < numbers[middle]:
      last  = _____   # (A) middle - 1   (B) middle + 1   (C) first
    else:
      first = _____   # (A) middle + 1   (B) middle - 1   (C) 0
```

**13. Συμπλήρωση κώδικα Ταξινόμησης σε Αύξουσα Σειρά:**

```
if numbers[j] _____ numbers[j-1]:    # (A) <   (B) >   (C) ==
   numbers[j], numbers[j-1] = _____

# (A) numbers[j-1], numbers[j]   (B) numbers[j], numbers[j-1]    (C) numbers[i]
```

**14. Συμπλήρωση κώδικα Παράλληλης Ταξινόμησης (10 μονάδες):**

```
grades = [14,18,15,18,14,15,16,18]
names  = ["JAMES","EMMA","JOHN","OLIVIA","MICHAEL","ISABELLA","AVA","THOMAS"]

n = len(grades)

for i in range(n-1):
   for j in range(n-1, i, -1):
      if grades[j] > grades[j-1] or (grades[j] == grades[j-1] and _____ ):
         # (A) names[j] < names[j-1]     (B) names[j] > names[j-1]     (C) grades[j] <
grades[j-1]

         grades[j], grades[j-1] = _____       # (A) grades[j-1], grades[j]   (B) 0   (C) j+1
         names[j],  names[j-1]  = _____       # (A) names[j-1], names[j]     (B) names[i]   (C) ""
```