ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΕΠΛ131 Αρχές Προγραμματισμού

Ακαδημαϊκό Έτος 2016/17 – Εαρινό Εξάμηνο

ΤΕΛΙΚΗ ΕΞΕΤΑΣΗ ΕΞΑΜΗΝΟΥ

ΗΜΕΡΟΜΗΝΙΑ:	8 Μαΐου 2017
ΔΙΑΡΚΕΙΑ:	12:30μμ – 3:00μμ
ΑΙΘΟΥΣΑ:	Κτήριο ΧΩΔ01, Αίθουσα 108
ΔΙΔΑΣΚΟΥΣΑ:	Ελπίδα Κεραυνού-Παπαηλιού

Απαντήστε και τις τρεις ερωτήσεις
Κάθε ερώτηση λαμβάνει 33 μονάδες.

Ερώτηση 1

(α) Κατασκευάστε το πρόγραμμα **ThreeDigits.java**, το οποίο λαμβάνει από τη γραμμή εντολής τρία δεκαδικά ψηφία και στη συνέχεια παρουσιάζει το μικρότερο (min) και μεγαλύτερο (max) φυσικό αριθμό που μπορεί να δημιουργηθεί από τα εν λόγω τρία ψηφία καθώς και όλους τους φυσικούς αριθμούς από το σύνολο {min, …., max} που είναι παλινδρομικοί αριθμοί δηλαδή διαβάζονται το ίδιο και από τα αριστερά, και από τα δεξιά. Τους παλινδρομικούς αριθμούς τους παρουσιάζει ανά δέκα σε κάθε γραμμή. Πιο κάτω δίνονται κάποια παραδείγματα χρήσης του προγράμματος.

```
$ java ThreeDigits 3 2 1
Min = 123 Max = 321
The numbers that are palindromes are:
  131  141  151  161  171  181  191  202  212  222
  232  242  252  262  272  282  292  303  313
```

```
$ java ThreeDigits 0 8 0
Min = 8 Max = 800
The numbers that are palindromes are:
    8    9   11   22   33   44   55   66   77   88
   99  101  111  121  131  141  151  161  171  181
  191  202  212  222  232  242  252  262  272  282
  292  303  313  323  333  343  353  363  373  383
  393  404  414  424  434  444  454  464  474  484
  494  505  515  525  535  545  555  565  575  585
  595  606  616  626  636  646  656  666  676  686
  696  707  717  727  737  747  757  767  777  787
  797
```

(β) Κατασκευάστε το πρόγραμμα **FiveFSS.java** το οποίο λαμβάνει από τη γραμμή εντολής ένα έτος καθώς και την πρώτη μέρα του εν λόγω έτους (1 για Δευτέρα, 2 για Τρίτη, …, 7 για Κυριακή) και παρουσιάζει τους μήνες του συγκεκριμένου έτους που έχουν πέντε Παρασκευές, πέντε Σάββατα και πέντε Κυριακές. Ουσιαστικά ένας μήνας δεδομένου έτους για να ικανοποιεί την πιο πάνω συνθήκη πρέπει να αρχίζει Παρασκευή και να έχει 31 μέρες. Οι μήνες Απρίλιος, Ιούνιος, Σεπτέμβριος και Νοέμβριος έχουν 30 μέρες, ο δε Φεβρουάριος έχει 28 μέρες αν το έτος δεν είναι δίσεκτο, διαφορετικά έχει 29 μέρες, όπου ένα έτος είναι δίσεκτο εάν διαιρείται ακριβώς με το 4 και σε περίπτωση που διαιρείται ακριβώς με το 100, πρέπει να διαιρείται ακριβώς και με το 400. Οι υπόλοιποι μήνες έχουν 31 μέρες. Πιο κάτω δίνονται κάποια παραδείγματα χρήσης του προγράμματος:

```
$ java FiveFSS 1993 5
Months in year 1993 with five Fri, Sat and Sun:
January
October

$ java FiveFSS 2017 7
Months in year 2017 with five Fri, Sat and Sun:
December

$ java FiveFSS 2020 3
Months in year 2020 with five Fri, Sat and Sun:
May
```

(γ) Κατασκευάστε το πρόγραμμα **Ariadne.java** το οποίο λαμβάνει δύο φυσικούς αριθμούς από τη γραμμή εντολής, έστω x και y, όπου x > 99, y > 99 και x < y, και παρουσιάζει, ανά δέκα σε κάθε γραμμή, τους αριθμούς από το σύνολο {x, .., y} των οποίων το άθροισμα του πρώτου και τελευταίου τους ψηφίου, ισούται με το άθροισμα των υπολοίπων (δηλαδή των ενδιάμεσων) ψηφίων τους. Πιο κάτω δίνονται κάποια παραδείγματα χρήσης του προγράμματος:

```
$ java Ariadne 1000 1030
The ariadne nos from 1000 to 1030 are:
  1010  1021

$ java Ariadne 100 500
The ariadne nos from 100 to 500 are:
   110   121   132   143   154   165   176   187   198   220
   231   242   253   264   275   286   297   330   341   352
   363   374   385   396   440   451   462   473   484   495
```

```
$ java Ariadne 1000 1500
The ariadne nos from 1000 to 1500 are:
  1010  1021  1032  1043  1054  1065  1076  1087  1098  1100
  1111  1122  1133  1144  1155  1166  1177  1188  1199  1201
  1212  1223  1234  1245  1256  1267  1278  1289  1302  1313
  1324  1335  1346  1357  1368  1379  1403  1414  1425  1436
  1447  1458  1469

$ java Ariadne 10000 10500
The ariadne nos from 10000 to 10500 are:
 10010 10021 10032 10043 10054 10065 10076 10087 10098 10100
 10111 10122 10133 10144 10155 10166 10177 10188 10199 10201
 10212 10223 10234 10245 10256 10267 10278 10289 10302 10313
 10324 10335 10346 10357 10368 10379 10403 10414 10425 10436
 10447 10458 10469

$ java Ariadne 10 100
Wrong Input .. program terminates

$ java Ariadne 121 45
Wrong Input .. program terminates
```

Ερώτηση 2

Η δεύτερη ερώτηση αφορά την ανάπτυξη ενός αντικειμενοστραφούς συστήματος σε Java για την κατανομή θέσεων σε προπτυχιακά προγράμματα σπουδών **(EntranceSystem.java**). Τα εμπλεκόμενα αντικείμενα είναι ο υποψήφιος (**Candidate**) και το πρόγραμμα σπουδών (**Program**) οι κλάσεις των οποίων ορίζονται μερικώς πιο κάτω:

```
public class Candidate { // Η κλάση αντικειμένου Candidate

 private String id; // η ταυτότητα του υποψηφίου
 private double score; // ο βαθμός κατάταξής του
 private String[] preferences; // οι προτιμήσεις του υποψηφίου
   // για πέντε προγράμματα σπουδών, σε φθίνουσα σειρά προτίμησης –
   // η κάθε προτίμηση είναι ο κωδικός του σχετικού
   // προγράμματος σπουδών
 private boolean[] activePrefs = // ποιες προτιμήσεις του
                                 // υποψηφίου είναι ενεργές
    {true, true, true, true, true}; // αρχικά είναι όλες ενεργές
 private boolean allocationMade = false; // αρχικά δεν του έχει
                                         // δοθεί καμία θέση φοίτησης
 private String studyPlace; // όταν πάρει θέση φοίτησης,
        // καταχωρείται και ο κωδικός του συγκεκριμένου προγράμματος

 public Candidate(String identity, double s, String[] ps){
   id = identity;
   score = s;
   preferences = ps;
 }

 public String getId(){ return id; }

 public double getScore(){ return score; }
```

```
 public boolean hasAllocation() { return allocationMade; }

 // Κατανομή θέσης στον υποψήφιο στο πρόγραμμα με κωδικό prog
 public void makeAllocation (String prog){
  allocationMade = true;
  studyPlace = prog;
 }

 public String getPreference (int n) { // η n-οστή προτίμηση
      return preferences[n-1]; }

 // Μετατροπή προτιμήσεων σε εκτυπώσιμη μορφή
 public String toStringPrefs (){
  String s = "\nCandidate " + id +
             " has the following preferences:\n";
  for (int i = 0; i < preferences.length; i++){
        s = s + "\t>> Program " + preferences[i] +
            " is choice " + (i+1) + "\n";
  }
  return s;
 }

// Μετατροπή υποψηφίου σε εκτυπώσιμη μορφή – ΝΑ ΟΡΙΣΘΕΙ
public String toString(){
     . . . . . .
 }

 // Καταχώρηση υποψηφίου στον κατάλογο υποψηφίων (μέσω της
 // μεθόδου αναφοράς insertCandidate του αντικειμένου Program)
 // για κάθε πρόγραμμα το οποίο συνιστά μια από τις πέντε
 // προτιμήσεις του υποψηφίου. Το σύνολο των προγραμμάτων
 // δίνεται στην παράμετρο ps – ΝΑ ΟΡΙΣΘΕΙ

 public void insertPrefs (Program[] ps){
     . . . . . .
 }

 // Το πρόγραμμα με κωδικό prog αποτελεί μια από τις
 // προτιμήσεις του υποψηφίου για την οποία εξασφαλίζει θέση.
 // Η εν λόγω θέση μπορεί μόνο να βελτιωθεί. Συνεπώς ο
 // υποψήφιος απενεργοποιεί την όποια ενεργή, χαμηλότερη,
 // προτίμησή του και αφαιρεί τον εαυτό του από τους
 // καταλόγους ενδιαφερόμενων υποψηφίων των σχετικών
 // προγραμμάτων (μέσω της μεθόδου removeCandidate του αντικειμένου
 // Program). Το σύνολο των προγραμμάτων δίνεται στην παράμετρο ps.
 // Αν γίνει τουλάχιστο μια τέτοια «αφαίρεση» του εν λόγω υποψηφίου
 // η μέθοδος επιστρέφει true, διαφορετικά επιστρέφει false
 // – ΝΑ ΟΡΙΣΘΕΙ

 public boolean releasePrefsBelow(String prog, Program[] ps){
     . . . . . .
 }

// Η main() δεν χρησιμοποιείται
 public static void main (String[] args){}
}
```

```
public class Program { // Η κλάση αντικειμένου Program

 private String code; // ο κωδικός του προγράμματος σπουδών
 private int places;  // ο αριθμός θέσεων φοίτησης στο πρόγραμμα
 private int count = 0; // το πλήθος των ενδιαφερόμενων υποψηφίων
                        // – αρχικά 0
 private Candidate[] cands; // ο κατάλογος των ενδιαφερόμενων
       // υποψηφίων για το πρόγραμμα – ουσιαστικά οι θυρίδες από το
       // 0 μέχρι το count-1, ταξινομημένες σε φθίνουσα σειρά των
       // βαθμών κατάταξης των εν λόγω ενδιαφερόμενων υποψηφίων
 private int freePlaces; // πόσες θέσεις του προγράμματος σπουδών
              // παραμένουν ελεύθερες μετά την ολοκλήρωση της
              // κατανομής θέσεων σε όλα τα προγράμματα σπουδών

 public Program(String c, int ps, int cs){
   code = c;
   places = ps;
   freePlaces = ps;
   cands = new Candidate[cs];
 }

 public String getCode(){return code;}

 public int getPlaces(){return places;}

 public int getCount(){return count;}

 public Candidate[] getCands(){return cands;}

 public int getFreePlaces(){return freePlaces;}

 private static int min(int x, int y){
   if (x < y) return x; else return y;
 }

 // Μετατροπή καταλόγου υποψηφίων σε εκτυπώσιμη μορφή
 public String toStringCands(){
   String s = "The ranked list of candidates is";
   for (int i = 0; i <  count; i++){
      s = s + "\n>> " + cands[i];
   }
   return s;
 }

 // Μετατροπή της κατανομής ενδιαφερόμενων υποψηφίων που πήραν θέση
 // στο πρόγραμμα σε εκτυπώσιμη μορφή
 public String toStringAllocations(){
   String s =
     "The following candidates get a place on the programme";
   for (int i = 0; i <  min(places,count); i++){
      s = s + "\n>> " + cands[i].getId() + "  " +
          cands[i].getScore();
   }
   s = s + "\nThe programme has " + freePlaces +
       " free places";
   return s;
 }
```

```
// Μετατροπή προγράμματος σε εκτυπώσιμη μορφή, όπου παρουσιάζονται
// ο αριθμός των θέσεών του, οι υποψήφιοι που πήραν θέση στο
// πρόγραμμα και πόσες ελεύθερες θέσεις έχουν μείνει – ΝΑ ΟΡΙΣΘΕΙ

public String toString(){
   . . . . . .
}

 // Η μέθοδος allocatePlaces κατανέμει τις θέσεις του προγράμματος
 // στους πρώτους σε σειρά ενδιαφερόμενους υποψηφίους (μέσω της
 // μεθόδου makeAllocation του αντικειμένου Candidate) – ΝΑ ΟΡΙΣΘΕΙ

public void allocatePlaces(){
    . . . . . .
}

// Καταχώρηση υποψηφίου c στον κατάλογο των ενδιαφερόμενων υποψηφίων
// του προγράμματος, έτσι ώστε ο κατάλογος να είναι ταξινομημένος
// σε φθίνουσα σειρά των βαθμών κατάταξης των ενδιαφερόμενων
// υποψηφίων - ΝΑ ΟΡΙΣΘΕΙ

public void insertCandidate(Candidate c){
   . . . . . .
 }

// Διαγραφή του υποψηφίου με ταυτότητα identity από τον κατάλογο των
// ενδιαφερόμενων υποψηφίων του προγράμματος – ΝΑ ΟΡΙΣΘΕΙ

public void removeCandidate(String identity){
   . . . . . .
 }

// Το πρόγραμμα σπουδών ζητεί από τον κάθε υποψήφιο στον οποίο
// διασφαλίζεται θέση φοίτησης στο εν λόγω πρόγραμμα (δηλαδή οι
// ενδιαφερόμενοι υποψήφιοι που καταλαμβάνουν πρώτοι σε σειρά το
// σχετικό αριθμό θέσεων φοίτησης) να «απελευθερώσει» (μέσω της
// μεθόδου releasePrefsBelow του αντικειμένου Candidate)τυχόν θέσεις
// χαμηλότερης προτίμησης που κρατεί σε άλλα προγράμματα σπουδών. Το
// σύνολο των προγραμμάτων δίνεται στην παράμετρο ps. Η τιμή εξόδου
// της μεθόδου επισημαίνει κατά πόσο έχει γίνει τουλάχιστο μια
// «απελευθέρωση» θέσης – ΝΑ ΟΡΙΣΘΕΙ

public boolean requestPrefsRelease(Program[] ps){
  . . . . . .
}

 public static void main(String[] args){}

}
```

Ορίστε τις μεθόδους αναφοράς που υποδεικνύονται πιο πάνω, δηλαδή για το αντικείμενο Candidate τις μεθόδους αναφοράς toString, insertPrefs και releasePrefsBelow, και για το αντικείμενο Program τις μεθόδους αναφοράς toString, allocatePlaces, insertCandidate, removeCandidate και requestPrefsRelease.

Για καλύτερη κατανόηση του όλου συστήματος σας δίνετε στη συνέχεια πλήρως ορισμένη η client κλάση EntranceSystem, μαζί με το αποτέλεσμα της εκτέλεσης του προγράμματος σε σχέση με τα στοιχεία που δίνονται στο αρχείο κειμένου Cands.txt, το περιεχόμενο του οποίου επίσης σας δίνετε.

Δώστε μια συνοπτική περιγραφή της λειτουργίας του προγράμματος EntranceSystem καθώς και της «δομής» του αρχείου κειμένου από το οποίο, μέσω file redirection, αντλεί τα στοιχεία του το πρόγραμμα.

Αρχείο Cands.txt

```
7 31
P1 5
P2 4
P3 5
P4 4
P5 5
P6 5
P7 5
C1 16.25 P2 P5 P1 P6 P7
C2 20.0 P4 P5 P2 P7 P6
C3 18.36 P2 P5 P1 P4 P7
C4 19.56 P3 P4 P5 P1 P6
C5 15.78 P5 P2 P1 P3 P6
C6 12.78 P1 P2 P3 P4 P5
C7 15.76 P3 P1 P5 P2 P6
C8 17.50 P4 P5 P3 P7 P1
C9 15.67 P1 P3 P5 P7 P6
C10 19.12 P1 P3 P5 P4 P7
C11 15.45 P1 P2 P3 P4 P5
C12 16.34 P3 P2 P4 P5 P1
C13 19.67 P5 P4 P1 P3 P7
C14 18.23 P3 P2 P4 P1 P6
C15 12.80 P1 P5 P3 P4 P7
C16 10.67 P1 P4 P2 P7 P6
C17 17.06 P6 P3 P5 P1 P4
C18 18.56 P7 P5 P2 P3 P1
C19 10.50 P5 P4 P6 P1 P2
C20 19.05 P5 P1 P2 P4 P3
C21 16.45 P4 P3 P2 P1 P6
C22 15.78 P4 P3 P6 P5 P1
C23 12.67 P5 P4 P2 P7 P1
C24 18.23 P3 P4 P2 P1 P5
C25 13.45 P6 P5 P7 P4 P1
C26 16.12 P4 P6 P5 P7 P1
C27 17.23 P4 P3 P5 P2 P1
C28 16.78 P5 P4 P3 P6 P2
C29 13.47 P2 P4 P1 P5 P7
C30 19.30 P1 P3 P5 P4 P6
C31 10.05 P1 P2 P3 P4 P5
```

```
public class EntranceSystem { //Client class EntranceSystem

public static void displayCandAllocations (Candidate[] cands){
   System.out.print("\n\nCANDIDATE ALLOCATIONS\n");
   for (int i = 0; i < cands.length; i++)
      System.out.println(cands[i]);
   System.out.print("\n\n");
 }

public static void displayProgAllocations (Program[] progs){
   System.out.print("\n\nPROGRAM ALLOCATIONS\n");
   for (int i = 0; i < progs.length; i++)
      System.out.println(progs[i]);
   System.out.print("\n\n");
 }

 public static void main(String[] args){
  int P = StdIn.readInt();
  int C = StdIn.readInt();
  Program[] progs = new Program[P];
  Candidate[] cands = new Candidate[C];

  for (int i = 0; i < P; i++){
   String code = StdIn.readString();
   int places = StdIn.readInt();
   progs[i] = new Program(code, places, C);
 }

 for (int i = 0; i < C; i++){
   String Id = StdIn.readString();
   double score = StdIn.readDouble();
   String[] prefs = new String[5];
   for (int j = 0; j < 5; j++)
      prefs[j] = StdIn.readString();
   cands[i] = new Candidate(Id, score, prefs);
 }

 for (int i = 0; i < C; i++){
   cands[i].insertPrefs(progs);
 }

 boolean releaseMade;
 do {
      releaseMade = false;
      for (int i = 0; i < P; i++) {
         boolean res = progs[i].requestPrefsRelease(progs);
         releaseMade = releaseMade || res;
      }
 } while(releaseMade);

 for (int i = 0; i < P; i++)
      progs[i].allocatePlaces();

  displayCandAllocations(cands);
  displayProgAllocations(progs);
 }
}
```

Παράδειγμα εκτέλεσης προγράμματος

```
$ java EntranceSystem < Cands.txt

CANDIDATE ALLOCATIONS
C1 with score 16.25 has been allocated a place on P2
C2 with score 20.0 has been allocated a place on P4
C3 with score 18.36 has been allocated a place on P2
C4 with score 19.56 has been allocated a place on P3
C5 with score 15.78 has been allocated a place on P5
C6 with score 12.78 has been allocated a place on P2
C7 with score 15.76 has been allocated a place on P1
C8 with score 17.5 has been allocated a place on P4
C9 with score 15.67 has been allocated a place on P1
C10 with score 19.12 has been allocated a place on P1
C11 with score 15.45 has been allocated a place on P1
C12 with score 16.34 has been allocated a place on P3
C13 with score 19.67 has been allocated a place on P5
C14 with score 18.23 has been allocated a place on P3
C15 with score 12.8 has been allocated a place on P5
C16 with score 10.67 has been allocated a place on P7
C17 with score 17.06 has been allocated a place on P6
C18 with score 18.56 has been allocated a place on P7
C19 with score 10.5 has been allocated a place on P6
C20 with score 19.05 has been allocated a place on P5
C21 with score 16.45 has been allocated a place on P4
C22 with score 15.78 has been allocated a place on P3
C23 with score 12.67 has been allocated a place on P7
C24 with score 18.23 has been allocated a place on P3
C25 with score 13.45 has been allocated a place on P6
C26 with score 16.12 has been allocated a place on P6
C27 with score 17.23 has been allocated a place on P4
C28 with score 16.78 has been allocated a place on P5
C29 with score 13.47 has been allocated a place on P2
C30 with score 19.3 has been allocated a place on P1
C31 with score 10.05 has not been allocated a place

PROGRAM ALLOCATIONS
Program P1 has 5 places
The following candidates get a place on the programme
>> C30  19.3
>> C10  19.12
>> C7  15.76
>> C9  15.67
>> C11  15.45
The programme has 0 free places
Program P2 has 4 places
The following candidates get a place on the programme
>> C3  18.36
>> C1  16.25
>> C29  13.47
>> C6  12.78
The programme has 0 free places
Program P3 has 5 places
The following candidates get a place on the programme
```

```
>> C4  19.56
>> C24  18.23
>> C14  18.23
>> C12  16.34
>> C22  15.78
The programme has 0 free places
Program P4 has 4 places
The following candidates get a place on the programme
>> C2  20.0
>> C8  17.5
>> C27  17.23
>> C21  16.45
The programme has 0 free places
Program P5 has 5 places
The following candidates get a place on the programme
>> C13  19.67
>> C20  19.05
>> C28  16.78
>> C5  15.78
>> C15  12.8
The programme has 0 free places
Program P6 has 5 places
The following candidates get a place on the programme
>> C17  17.06
>> C26  16.12
>> C25  13.45
>> C19  10.5
The programme has 1 free places
Program P7 has 5 places
The following candidates get a place on the programme
>> C18  18.56
>> C23  12.67
>> C16  10.67
The programme has 2 free places
```

Ερώτηση 3

Απαντήστε είτε το μέρος (α) είτε το μέρος (β) της ερώτησης.

(α) Κατασκευάστε το πρόγραμμα **Format.java** το οποίο επεξεργάζεται αρχεία κειμένου, για παράδειγμα αρχεία όπως το ακόλουθο cs.txt:

Computer programming (often shortened to programming) is a process that leads from an original formulation of a computing problem to executable computer programs. Programming involves activities such as analysis, developing understanding, generating algorithms, verification of requirements of algorithms including their correctness and resources consumption, and implementation (commonly referred to as coding[1][2]) of algorithms in a target programming language. Source code is written in one or more programming languages. The purpose of programming is to find a sequence of instructions that will automate performing a specific task or solving a given problem. The process of programming thus often requires expertise in many different subjects, including knowledge of the application domain, specialized algorithms, and formal logic.

Η επεξεργασία που κάνει το πρόγραμμα διαφαίνεται μέσω των ακόλουθων τριών παραδειγμάτων χρήσης του.

Στο ακόλουθο **πρώτο παράδειγμα**, οι λέξεις τοποθετούνται σε γραμμές, το μήκος των οποίων δεν υπερβαίνει τους 40 χαρακτήρες (δεύτερο όρισμα στη γραμμή εντολής), ενώ η ευθυγράμμιση είναι μόνο από τα αριστερά (πρώτο όρισμα στη γραμμή εντολής, −l):

```
$ java Format -l 40 < cs.txt
Computer programming (often shortened
to programming) is a process that leads
from an original formulation of a
computing problem to executable
computer programs. Programming involves
activities such as analysis, developing
understanding, generating algorithms,
verification of requirements of
algorithms including their correctness
and resources consumption, and
implementation (commonly referred to as
coding[1][2]) of algorithms in a target
programming language. Source code is
written in one or more programming
languages. The purpose of programming
is to find a sequence of instructions
that will automate performing a
specific task or solving a given
problem. The process of programming
thus often requires expertise in many
different subjects, including knowledge
of the application domain, specialized
algorithms, and formal logic.
```

Στο ακόλουθο **δεύτερο παράδειγμα**, οι λέξεις τοποθετούνται σε γραμμές, το μήκος των οποίων δεν υπερβαίνει τους 60 χαρακτήρες (δεύτερο όρισμα στη γραμμή εντολής), και επιπρόσθετα υπάρχει ευθυγράμμιση και από τα δεξιά (πρώτο όρισμα στη γραμμή εντολής, −r) πλην της τελευταίας γραμμής:

```
$ java Format -r 60 < cs.txt
Computer programming (often shortened to programming) is a
process that leads from an original formulation of a
computing problem to executable computer programs.
Programming involves activities such as analysis,
developing understanding, generating algorithms,
verification of requirements of algorithms including their
correctness and resources consumption, and implementation
(commonly referred to as coding[1][2]) of algorithms in a
target programming language. Source code is written in one
or more programming languages. The purpose of programming
is to find a sequence of instructions that will automate
```

```
performing  a specific task or solving a given problem. The
process  of  programming  thus  often requires expertise in
many  different  subjects,  including  knowledge  of  the
application  domain,  specialized  algorithms,  and  formal
logic.
```

Στο ακόλουθο **τρίτο παράδειγμα**, οι λέξεις τοποθετούνται σε γραμμές, το μήκος των οποίων δεν υπερβαίνει τους 50 χαρακτήρες (δεύτερο όρισμα στη γραμμή εντολής), και επιπρόσθετα παρουσιάζονται centralized (πρώτο όρισμα στη γραμμή εντολής, −c) σε σχέση με ένα μήκος 65 χαρακτήρων (τρίτο όρισμα στη γραμμή εντολής):

```
$ java Format -c 50 65 < cs.txt
           Computer programming (often shortened to
          programming) is a process that leads from an
         original formulation of a computing problem to
           executable computer programs. Programming
        involves activities such as analysis, developing
             understanding, generating algorithms,
           verification of requirements of algorithms
           including their correctness and resources
           consumption, and implementation (commonly
        referred to as coding[1][2]) of algorithms in a
          target programming language. Source code is
       written in one or more programming languages. The
        purpose of programming is to find a sequence of
          instructions that will automate performing a
         specific task or solving a given problem. The
           process of programming thus often requires
        expertise in many different subjects, including
        knowledge of the application domain, specialized
                 algorithms, and formal logic.
```

Συνεπώς, υπάρχουν τρεις διαφορετικές επεξεργασίες που μπορούν να γίνουν σε δεδομένα αρχεία κειμένου. Το ποια επεξεργασία θα γίνει προσδιορίζεται ως πρώτο όρισμα στη γραμμή εντολής (–l, –r, ή –c). Στην περίπτωση των επεξεργασιών –l και –r ακολουθεί μόνο ένα δεύτερο όρισμα που αντίστοιχα δίνει το μέγιστο μήκος, και το μήκος, των γραμμών. Στην περίπτωση της επεξεργασίας –c ακολουθούν δύο άλλα ορίσματα που προσδιορίζουν το μέγιστο μήκος των γραμμών, και το μήκος του πλαισίου ως προς το centralization.

Ορίστε το πρόγραμμα Format.java, βάσει των πιο πάνω προδιαγραφών, και επίσης απλοποιείστε τις ακόλουθες εντολές:

```
$ java Format -r 60 < cs.txt | java Format -r 45

$ java Format -c 35 45 < cs.txt | java Format -l 40 | java Format -c 45 60
```

(β) Κατασκευάστε το πρόγραμμα **Grep.java** το οποίο ανακτά τις γραμμές δεδομένου αρχείου κειμένου, οι οποίες περιλαμβάνουν (συζευκτικά ή διαζευκτικά) ή δεν περιλαμβάνουν, συγκεκριμένες ακολουθίες χαρακτήρων (strings). Τα ορίσματα της σχετικής αναζήτησης προσδιορίζονται στη γραμμή εντολής, όπως δεικνύεται μέσω των ακόλουθων παραδειγμάτων χρήσης του προγράμματος, σε σχέση και πάλι με το αρχείο cs.txt, το οποίο δίνεται και πάλι πιο κάτω:

Computer programming (often shortened to programming) is a process that leads
from an original formulation of a computing problem to executable computer
programs. Programming involves activities such as analysis, developing
understanding, generating algorithms, verification of requirements of
algorithms including their correctness and resources consumption, and implementation
(commonly referred to as coding[1][2]) of algorithms in a target programming language.
Source code is written in one or more programming languages. The purpose of programming
is to find a sequence of instructions that will automate performing a specific
task or solving a given problem. The process of programming thus often requires
expertise in many different subjects, including knowledge of the application domain,
specialized algorithms, and formal logic.

Στο ακόλουθο **πρώτο παράδειγμα**, ο χρήστης ζητά τις γραμμές του αρχείου που περιλαμβάνουν είτε τη λέξη programming, είτε τη λέξη language (μπορεί και τις δύο), καθώς και τη λέξη process. Μόνο δύο γραμμές του αρχείου cs.txt ικανοποιούν την εν λόγω εντολή αναζήτησης, οι οποίες παρουσιάζονται:

$ java Grep -o programming language -a process < cs.txt
Computer programming (often shortened to programming) is a process that leads
task or solving a given problem. The process of programming thus often requires

Στο ακόλουθο **δεύτερο παράδειγμα**, ο χρήστης ζητά τις γραμμές του αρχείου που περιλαμβάνουν τη λέξη algorithms, αλλά δεν περιλαμβάνουν τις λέξεις coding ή code. Τρεις γραμμές του αρχείου cs.txt ικανοποιούν την εν λόγω εντολή αναζήτησης, οι οποίες παρουσιάζονται:

$ java Grep -a algorithms -n coding code < cs.txt
understanding, generating algorithms, verification of requirements of
algorithms including their correctness and resources consumption, and implementation
specialized algorithms, and formal logic.

Στο ακόλουθο **τρίτο παράδειγμα**, ο χρήστης ζητά τις γραμμές του αρχείου που περιλαμβάνουν είτε τη λέξη knowledge, είτε τη λέξη programming (μπορεί και τις δύο). Οι γραμμές του αρχείου cs.txt που ικανοποιούν την εν λόγω εντολή αναζήτησης παρουσιάζονται:

$ java Grep -o knowledge programming < cs.txt
Computer programming (often shortened to programming) is a process that leads
(commonly referred to as coding[1][2]) of algorithms in a target programming language.
Source code is written in one or more programming languages. The purpose of programming
task or solving a given problem. The process of programming thus often requires
expertise in many different subjects, including knowledge of the application domain,

Συνεπώς, η εντολή αναζήτησης προσδιορίζεται στη γραμμή εντολής μέσω των flags –a, −o, και –n, που αντίστοιχα υποδηλώνουν «και» (and), «ή» (or) και «όχι» (not). Οι λέξεις (μία ή περισσότερες) που ακολουθούν το flag –a πρέπει να εμφανίζονται όλες στην κάθε γραμμή που ανακτάται, οι λέξεις που ακολουθούν το flag –n (μία ή περισσότερες) δεν πρέπει να εμφανίζονται στις γραμμές που ανακτώνται, ενώ από τις λέξεις που ακολουθούν το flag –o (δύο ή περισσότερες), τουλάχιστο μία πρέπει να

εμφανίζεται στις γραμμές που ανακτώνται. Σε κάποια εντολή αναζήτησης, μπορεί να χρησιμοποιείται οποιοσδήποτε συνδυασμός flags. Αν δεν δοθεί κανένα flag, ουσιαστικά παρουσιάζεται πίσω αυτούσιο το αρχείο κειμένου όπως φαίνεται στο ακόλουθο παράδειγμα:

$ java Grep < cs.txt
Computer programming (often shortened to programming) is a process that leads
from an original formulation of a computing problem to executable computer
programs. Programming involves activities such as analysis, developing
understanding, generating algorithms, verification of requirements of
algorithms including their correctness and resources consumption, and implementation
(commonly referred to as coding[1][2]) of algorithms in a target programming language.
Source code is written in one or more programming languages. The purpose of programming
is to find a sequence of instructions that will automate performing a specific
task or solving a given problem. The process of programming thus often requires
expertise in many different subjects, including knowledge of the application domain,
specialized algorithms, and formal logic.

Σημειώνεται ότι η αναφορά σε «λέξεις» πιο πάνω σημαίνει «γραμματοσειρές» (strings). Ορίστε το πρόγραμμα Grep.java και επίσης απλοποιείστε τις ακόλουθες εντολές:

$ java Grep -o algorithms programming < cs.txt | java Grep -n code

$ java Grep –n search < cs.txt | java Grep –n computer | java Grep –a problem

ΤΕΛΟΣ ΕΡΩΤΗΣΕΩΝ

ΧΩΡΟΣ ΑΠΑΝΤΗΣΕΩΝ

Ονοματεπώνυμο Φοιτητή: ----------

Ταυτότητα: ----------

Υπογραφή: ----------

Ερωτήσεις	**Μονάδες**
Ερώτηση 1	
Ερώτηση 2	
Ερώτηση 3	
Σύνολο Μονάδων	

Παρατηρήσεις Διδάσκοντα

Απάντηση στην Ερώτηση 1

(τυχόν συνέχεια στην απάντηση της Ερώτησης 1)

(τυχόν συνέχεια στην απάντηση της Ερώτησης 1)

Απάντηση στην Ερώτηση 2

(τυχόν συνέχεια στην απάντηση της Ερώτησης 2)

(τυχόν συνέχεια στην απάντηση της Ερώτησης 2)

Απάντηση στην Ερώτηση 3

(τυχόν συνέχεια στην απάντηση της Ερώτησης 3)

(τυχόν συνέχεια στην απάντηση της Ερώτησης 3)

Επιπρόσθετο φύλλο (για συμπλήρωση απάντησης ή πρόχειρο)

Επιπρόσθετο φύλλο (για συμπλήρωση απάντησης ή πρόχειρο)

Επιπρόσθετο φύλλο (για συμπλήρωση απάντησης ή πρόχειρο)

Επιπρόσθετο φύλλο (για συμπλήρωση απάντησης ή πρόχειρο)

ΤΕΛΟΣ ΕΞΕΤΑΣΤΙΚΟΥ ΔΟΚΙΜΙΟΥ