ΕΙΣΑΓΩΓΗ ΣΤΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ (2009-10)

Άσκηση 1

Ένας θετικός ακέραιος αριθμός μεγαλύτερος του 1 ονομάζεται *πλήρης-δυνάμεων* (*powerful*), ή ΠΔ για τη συνέχεια, όταν αναλύοντάς τον σε γινόμενο δυνάμεων πρώτων παραγόντων, για κάθε πρώτο παράγοντά του, ο εκθέτης του είναι μεγαλύτερος του 1. Για παράδειγμα, το 9000 είναι ΠΔ αριθμός, αφού $9000 = 2^3 \times 3^2 \times 5^3$, ενώ το 279 δεν είναι, αφού $279 = 3^2 \times 31$ (το 31 έχει εκθέτη ίσο με 1).

Πιο μαθηματικά, αν ο αριθμός x αναλύεται σε πρώτους παράγοντες σαν

$$x = p_1^{e_1} \times p_2^{e_2} \times \ldots \times p_n^{e_n}$$

όπου τα p_i $(1 \leq i \leq n)$ είναι πρώτοι, ικανή και αναγκαία συνθήκη για να είναι ο x ΠΔ είναι να ισχύει $e_i > 1$, για κάθε i με $1 \leq i \leq n$.

Γράψτε ένα πρόγραμμα C (έστω ότι το πηγαίο αρχείο του ονομάζεται "`powerful.c`") το οποίο να επιλέγει με τυχαίο τρόπο 3000 θετικούς ακεραίους αριθμούς μεγαλύτερους του 1 (όχι κατ' ανάγκη διαφορετικούς μεταξύ τους) και να βρίσκει ποιοι απ' αυτούς είναι ΠΔ. Το πλήθος των αριθμών που θα ελεγχθούν (3000) να μην τοποθετηθεί απ' ευθείας μέσα στο πρόγραμμά σας, αλλά να ορισθεί μέσω `#define` σαν τιμή της συμβολικής σταθεράς `COMPUTATIONS`. Ο τυχαίος τρόπος επιλογής των αριθμών που θα ελεγχθούν πρέπει να ακολουθεί τη διαδικασία που περιγράφεται στη συνέχεια.

Στην C μπορούμε να "γεννήσουμε" τυχαίους αριθμούς με χρήση των συναρτήσεων `srand` και `rand`, όπως επιδεικνύεται στο πρόγραμμα `http://www.di.uoa.gr/~ip/cprogs/gcdlcm.c` (ή σελίδες 45–46 των διαφανειών του μαθήματος, στο `http://www.di.uoa.gr/~ip/K04.pdf`). Λεπτομέρειες για τη χρήση των συναρτήσεων `srand` και `rand` μπορείτε να μάθετε δίνοντας στα Suns του Τμήματος "`man srand`" ή "`man rand`" (πιθανότατα, σε άλλα συστήματα Unix, όπως για παράδειγμα στα μηχανήματα Linux του Τμήματος, να χρειάζεται "`man 3 srand`" ή "`man 3 rand`").

Έστω ότι γεννάτε έναν τυχαίο αριθμό y με χρήση της συνάρτησης `rand`. Ο αριθμός που θα πρέπει να ελέγξετε αν είναι ΠΔ ή όχι είναι ο $x = (y \bmod 32768) + 2$. Για την αρχικοποίηση της γεννήτριας των τυχαίων αριθμών, να χρησιμοποιήσετε την τρέχουσα ώρα, μέσω της συνάρτησης `time`[1], όπως ακριβώς γίνεται και στο πρόγραμμα `gcdlcm.c` που προαναφέρθηκε.

Το πρόγραμμά σας να εκτυπώνει τους αριθμούς εκείνους από τους 3000 (που είναι η τιμή της συμβολικής σταθεράς `COMPUTATIONS`) που θα ελέγξετε οι οποίοι είναι ΠΔ και, επίσης, να υπολογίζει και να εκτυπώνει το ποσοστό των ΠΔ αριθμών που βρέθηκαν.

Μία ενδεικτική εκτέλεση[2] του προγράμματος φαίνεται στη συνέχεια:

```
% ./powerful
Current time is 1256558419

18000 is powerful
 1521 is powerful
  961 is powerful
 2312 is powerful
13824 is powerful
```

[1] Πληροφοριακά, η τιμή που επιστρέφει η `time` ισούται με το πλήθος των δευτερολέπτων που έχουν περάσει από την 1/1/1970, ώρα 00:00, μέχρι τη στιγμή που κλήθηκε.

[2] Η συγκεκριμένη εκτέλεση έγινε σε μηχάνημα Sun του Τμήματος. Στα μηχανήματα Linux του Τμήματος, η εκτέλεση, για το δεδομένο "φύτρο" (1256558419) της γεννήτριας τυχαίων αριθμών, δίνει ποσοστό ΠΔ αριθμών ίσο με 1.0%. Σε άλλες πλατφόρμες (π.χ. PC/Dev-C++) τα αποτελέσματα ενδέχεται να διαφέρουν, διότι η γεννήτρια τυχαίων αριθμών μπορεί να παράγει διαφορετική ακολουθία αριθμών, ακόμα και με το ίδιο "φύτρο".

```
 4900 is powerful
  961 is powerful
 5408 is powerful
15876 is powerful
 8000 is powerful
10976 is powerful
 1024 is powerful
  972 is powerful
27556 is powerful
 3087 is powerful
30375 is powerful
19881 is powerful
   32 is powerful
17956 is powerful
26244 is powerful
19600 is powerful
13068 is powerful
   32 is powerful
  841 is powerful
 2916 is powerful
 2744 is powerful
25088 is powerful

Generated 3000 numbers - found 0.9% powerful
```

Η παράδοση της άσκησης αυτής συνίσταται στην υποβολή του πηγαίου αρχείου `powerful.c` με διαδικασία που θα ανακοινωθεί σύντομα.

Υπόδειξη: Σε πρώτη φάση, αγνοήστε την απαίτηση της εκφώνησης για την παραγωγή τυχαίων αριθμών και επικεντρωθείτε στη διατύπωση ενός αλγορίθμου, κατ΄ αρχήν σε ψευδογλώσσα και στη συνέχεια σε C, που επιλύει το πρόβλημα του ελέγχου αν ένας δεδομένος αριθμός είναι ΠΔ ή όχι. Το ποιον αριθμό θα ελέγχει το πρόγραμμα που θα γράψετε αρχικά μπορείτε να το ορίζετε μέσω `#define`. Αφού μετά από αρκετές δοκιμές βεβαιωθείτε ότι το πρόγραμμά σας δουλεύει σωστά, μπορείτε να το επεκτείνετε ώστε να κάνει τον έλεγχο για το σύνολο των αριθμών που θα γεννηθούν με τυχαίο τρόπο. Με άλλα λόγια, μην δοκιμάσετε να αντιμετωπίσετε ταυτόχρονα το βασικό αλγοριθμικό πρόβλημα της άσκησης και τα τεχνικά προβλήματα σχετικά με τη χρήση των `srand`, `rand` και `time`.