Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής & Υπολογιστών
https://courses.softlab.ntua.gr/pl2/

Γλώσσες Προγραμματισμού II

Αν δεν αναφέρεται διαφορετικά, οι ασκήσεις πρέπει να παραδίδονται στους διδάσκοντες σε ηλεκτρονική μορφή μέσω του συνεργατικού συστήματος ηλεκτρονικής μάθησης `moodle.softlab.ntua.gr`. Η προθεσμία παράδοσης θα τηρείται αυστηρά. Έχετε δικαίωμα να καθυστερήσετε το πολύ μία άσκηση.

Άσκηση 3 Το δις εξαμαρτείν ουκ ανδρός σοφού

Προθεσμία παράδοσης: 15/12/2019

Γράψτε δύο προγράμματα σε Haskell που να λύνουν το παρακάτω πρόβλημα. Το πρώτο θα πρέπει να χρησιμοποιεί αμιγώς συναρτησιακές (pure) δομές δεδομένων για τη λύση του προβλήματος. Το δεύτερο μπορεί να χρησιμοποιεί και μη συναρτησιακές (impure) δομές, όπως π.χ. πίνακες (arrays) σε κάποιο κατάλληλο monad. Υποβάλετε τις λύσεις σας (ξεχωριστά) στο σύστημα αυτόματης υποβολής και ελέγχου προγραμμάτων `grader.softlab.ntua.gr`.

Περιγραφή του προβλήματος. Σε ένα πρωτάθλημα online μπιλιάρδου, κάθε παίκτης παίζει με διάφορους άλλους παίκτες, με τη σειρά που ορίζει το σύστημα. Κάθε νίκη αποφέρει κάποιους βαθμούς και οι βαθμοί αυτοί αθροίζονται για κάθε παίκτη — αντίθετα οι ήττες δε δίνουν βαθμούς. Για να γίνει το παιχνίδι πιο ενδιαφέρον, οι διοργανωτές του πρωταθλήματος αποφάσισαν το εξής σύστημα βαθμολόγησης:

- Οι βαθμοί κάθε νίκης αυξάνουν ανάλογα με το πλήθος των διαδοχικών νικών που έχει πετύχει ένας παίκτης. Η πρώτη νίκη δίνει 1 βαθμό, η δεύτερη (συνεχόμενη) 2 βαθμούς, η τρίτη (πάλι συνεχόμενη) 4 βαθμούς, η τέταρτη 8 βαθμούς, κ.ο.κ.
- Σε περίπτωση ήττας τερματίζονται φυσικά οι διαδοχικές νίκες. Άρα, η πρώτη επόμενη νίκη θα δώσει πάλι 1 βαθμό.
- Αν ένας παίκτης χάσει δύο συνεχόμενες φορές, τότε αποκλείεται από το πρωτάθλημα (εξ ου και ο τίτλος του προβλήματος)

Έστω δύο φυσικοί αριθμοί $A \leq B$. Βρείτε με πόσους διαφορετικούς τρόπους μπορεί ένας παίκτης να συγκεντρώσει k βαθμούς, όπου $A \leq k \leq B$, προτού αποκλειστεί από το πρωτάθλημα.

Για παράδειγμα, έστω $A = 4$ και $B = 5$. Ένας παίκτης μπορεί να συγκεντρώσει 4 ή 5 βαθμούς με τους εξής δέκα τέσσερις (14) τρόπους, όπου "k" σημαίνει νίκη που δίνει k βαθμούς και "×" ήττα:

```
1 × 1 × 1 × 1 × ×
× 1 × 1 × 1 × 1 × ×
1 × 1 2 × ×
× 1 × 1 2 × ×
1 2 × 1 × ×
× 1 2 × 1 × ×
```

```
1 × 1 × 1 × 1 × 1 × ×
× 1 × 1 × 1 × 1 × 1 × ×
1 × 1 × 1 2 × ×
× 1 × 1 × 1 2 × ×
1 × 1 2 × 1 × ×
× 1 × 1 2 × 1 × ×
1 2 × 1 × 1 × ×
× 1 2 × 1 × 1 × ×
```

Είσοδος και έξοδος. Το πρόγραμμά σας θα διαβάζει τα δεδομένα από την τυπική είσοδο (stdin) και θα τυπώνει τα αποτελέσματα στην τυπική έξοδο (stdout).

Η πρώτη γραμμή της εισόδου περιέχει δύο ακέραιους αριθμούς N και M, χωρισμένους μεταξύ τους με ένα κενό διάστημα. Ο αριθμός N είναι το πλήθος των ερωτημάτων που ακολουθούν. Επειδή το πλήθος των δυνατών τρόπων συγκέντρωσης βαθμών μπορεί να είναι πολύ μεγάλος αριθμός, για κάθε ερώτημα σας ζητείται να εκτυπώσετε το υπόλοιπο της ακέραιας διαίρεσής του (modulo) με το δοθέντα αριθμό M. Κάθε μία από τις επόμενες N γραμμές περιέχει δύο ακέραιους αριθμούς A και B, χωρισμένους μεταξύ τους με ένα κενό διάστημα. Οι αριθμοί αυτοί αναπαριστούν ένα ερώτημα που πρέπει να απαντήσετε.

Η έξοδος πρέπει να αποτελείται από ακριβώς N γραμμές, κάθε μία από τις οποίες θα περιέχει ακριβώς έναν ακέραιο αριθμό: την απάντηση στο αντίστοιχο ερώτημα.

Παράδειγμα #1

Είσοδος	**Έξοδος**

`4 2019`
`2 4`
`3 3`
`4 5`
`17 42`
	`12`
	`4`
	`14`
	`936`

Παράδειγμα #2

Είσοδος	**Έξοδος**

`3 1000000007`
`0 10`
`1742 4217`
`500000 999999`
	`197`
	`376826466`
	`969790604`

Εξήγηση. Το 3ο ερώτημα του 1ου παραδείγματος ($A = 4, B = 5$) εξηγήθηκε παραπάνω. Για το 4ο ερώτημα του 1ου παραδείγματος ($A = 17, B = 42$), υπάρχουν $126.321.690$ τρόποι με τους οποίους μπορεί κάποιος να συγκεντρώσει μεταξύ 17 και 42 βαθμών. Επειδή $M = 2019$, η απάντηση είναι 936 (γιατί $126.321.690 = 62.566 \times 2019 + 936$).

Περιορισμοί.

- $1 \leq N \leq 100.000$
- $0 \leq A \leq B \leq 1.000.000$
- Όριο χρόνου εκτέλεσης: 3 sec. (pure) και 1 sec. (impure).
- Όριο μνήμης: 64 MB.

Για να βαθμολογηθεί με άριστα, η λύση σας πρέπει να είναι αποδοτική. Προσέξτε ότι στη Haskell το αποδοτικό διάβασμα της εισόδου, η χρήση αποδοτικών δομών δεδομένων και η αποφυγή της οκνηρίας (laziness) μπορεί να αποδειχθούν δύσκολο έργο.