Ο Αλγόριθμος του Ευκλείδη - gcd

Ο Ευκλείδης ήταν πολύ τυχερός. Μεγάλωσε το 300 π.Χ. στην Αλεξάνδρεια, το πολιτισμικό κέντρο της Μεσογείου — για κάποιους το λίκνο του μοντέρνου κόσμου — και έχει άμεση πρόσβαση στην ... Wikipedia της εποχής. Ο κόσμος γύρω του αλλάζει γρήγορα, η φιλοσοφία και τα μαθηματικά ανθούν και ο ίδιος δεν κάθεται με σταυρωμένα χέρια. Συγκεντρώνει όλα τα γνωστά μέχρι τότε μαθηματικά και συνθέτει τα "Στοιχεία", μια πραγματεία 13(!) τόμων η οποία 23 αιώνες αργότερα θα θεωρηθεί από τα σημαντικότερα κείμενα που έχουν γραφεί. Στις προτάσεις 1-2 του 7ου τόμου, ο Ευκλείδης περιγράφει έναν αλγόριθμο—η λέξη αλγόριθμος δεν θα εφευρεθεί για 1000+ ακόμα χρόνια — ο οποίος χρησιμοποιείται ακόμα και σήμερα. Προς τιμήν του, ο αλγόριθμος αυτός λέγεται Αλγόριθμος του Ευκλείδη. Ο αλγόριθμος του Ευκλείδη (Euclidean Algorithm) μας βοηθάει να λύσουμε το πρόβλημα του Μέγιστου Κοινού Διαιρέτη (ΜΚΔ) ή Greatest Common Divisor (GCD) στα Αγγλικά. Αν σας θυμίζει κάτι, έχετε καλή μνήμη· το πρόβλημα του ΜΚΔ το είχαμε ξαναδεί στο Δημοτικό! Τότε το λέγαμε "μου-κου-δου", αλλά επειδή οι περισσότεροι μάλλον το απωθήσαμε από την μνήμη μας (για προφανείς λόγους) προσφέρουμε εδώ μια υπενθύμιση: δοθέντων δύο αριθμών a και b ο μέγιστος κοινός διαιρέτης τους είναι ο μέγιστος αριθμός d ο οποίος διαιρεί τους a και b χωρίς να αφήνει υπόλοιπο. Η έκφραση "διαιρεί χωρίς να αφήνει υπόλοιπο" είναι τόσο κοινή που συχνά απλοποιείται σε "διαιρεί". Πιο επίσημα, έχουμε τους ορισμούς:

Ορισμός 1. Αν ο a και ο b είναι ακέραιοι με $a \neq 0$, λέμε ότι ο a διαιρεί τον b αν υπάρχει ακέραιος c έτσι ώστε b = a · c. Όταν ο a διαιρεί τον b λέμε ότι ο a είναι παράγοντας του b και ότι ο b είναι πολλαπλάσιο του a. Ο συμβολισμός a | b σημαίνει ότι ο a διαιρεί τον b (b mod a = 0). Αντίθετα a ∤ b συμβολίζει ότι ο a δεν διαιρεί τον b ($b \ mod \ a \neq 0$).

Ορισμός 2. Έστω ότι οι a και b είναι ακέραιοι, όχι μηδενικοί και οι δύο. Ο μεγαλύτερος ακέραιος d έτσι ώστε να είναι d | a και d | b ονομάζεται μέγιστος κοινός διαιρέτης των a και b και συμβολίζεται με gcd(a, b).

Για να δούμε πως μπορούμε να λύσουμε ένα τέτοιο πρόβλημα. Έστω ότι οι δύο ακέραιοι είναι: το 42 και το 18. Στο Δημοτικό, προκειμένου βρούμε το ΜΚΔ παραγοντοποιούσαμε τους δύο αριθμούς και ο ΜΚΔ ήταν ήταν το γινόμενο των κοινών 2 παραγόντων. Η παραγοντοποίηση του 18 είναι: 2 · 3 2 ενώ του 42 είναι 2 · 3 · 7 και επομένως οι 2 · 3 είναι κοινοί παράγοντες και gcd(42, 18) = 6. Η παραγοντοποίηση δεν είναι κακή μέθοδος αλλά είναι γενικά δύσκολη. Ο αλγόριθμος του Ευκλείδη προτείνει κάτι σχετικά πιο εύκολο1 :

$$\gcd(a,b) = \begin{cases} b & ,\text{if } a \bmod b = 0 \\ \gcd(b, a \bmod b) & ,\text{otherwise} \end{cases} \tag{1}$$

Για να "τρέξουμε" την αναδρομική Εξίσωση 1 για το παράδειγμά μας: gcd(42, 18). Έχουμε ότι 42 mod 18 = 6 (διάφορο του 0) και επομένως παίρνουμε τον δεύτερο κλάδο της συνάρτησης και

υπολογίζουμε το gcd(18, 42 mod 18) = gcd(18, 6). Τώρα όμως έχουμε ότι 18 mod 6 = 0 και επομένως από τον πρώτο κλάδο της συνάρτησης έχουμε: gcd(18, 6) = 6. Συνεπώς πήραμε και πάλι το αναμενόμενο αποτέλεσμα: gcd(42, 18) = 6. Εύκολο; Θα δείξει!

Για το ζητούμενο αυτής της άσκησης λοιπόν, καλείστε να γράψετε ένα πρόγραμμα gcd που υπολογίζει αυτόματα και αποδοτικά τον μέγιστο κοινό διαιρέτη δύο ακεραίων αριθμών χρησιμοποιώντας τον αναδρομικό αλγόριθμο του Ευκλείδη.

Τεχνικές Προδιαγραφές

• C Filepath: gcd/src/gcd.c
• Το πρόγραμμά θα πρέπει να παίρνει δύο ακεραίους αριθμούς στο δεκαδικό σύστημα ως ορίσματα από την γραμμή εντολών στην μορφή `./gcd num0 num1`. Αν το πρόγραμμα εκτελεστεί με ορίσματα που δεν ακολουθούν τις παραπάνω προδιαγραφές, πρέπει να εκτυπώσει αντίστοιχο μήνυμα όπως στα παρακάτω παραδείγματα και να επιστρέφει με κωδικό εξόδου (exit code) 1.
• Όλες οι παράμετροι θα είναι στο εύρος [−1018 , 1018]. Σε περίπτωση που δοθεί το μηδέν (0) το πρόγραμμά σας πρέπει να επιστρέφει με κωδικό εξόδου 1.
• Το αρχείο C που θα υποβληθεί πρέπει να μεταγλωττίζεται χωρίς ειδοποιήσεις για λάθη και με κωδικό επιστροφής (exit code) που να είναι 0. Συγκεκριμένα, το αρχείο σας πρέπει να μπορεί να μεταγλωττιστεί επιτυχώς με την ακόλουθη εντολή: `gcc -O3 -Wall -Wextra -Werror -pedantic -o gcd gcd.c`
• README Filepath: gcd/README.md 1Να αναφέρουμε ότι ο αυθεντικός αλγόριθμος του Ευκλείδη ήταν διαφορετικός [9], εδώ χρησιμοποιούμε μια πιο μοντέρνα εκδοχή του. 3
• Ένα αρχείο που να περιέχει στοιχεία εισόδου και ένα εξόδου διαφορετικά από αυτά της εκφώνησης. Συγκεκριμένα προτείνουμε να βάλετε έναν συνδυασμό που θεωρείται ότι είναι δύσκολος να γίνει σωστός.
– input Filepath: gcd/test/input.txt
– output Filepath: gcd/test/output.txt
Παράδειγμα που όμως δεν θα γίνει δεκτό από την άσκηση επειδή είναι ήδη στα παραδείγματα παρακάτω, για το input.txt: "942 1042" και για το output.txt: "2".

• Πρέπει να ολοκληρώνει την εκτέλεση μέσα σε: 1 δευτερόλεπτο.
Παρακάτω παραθέτουμε την αλληλεπίδραση με μια ενδεικτική λύση:

```
$ hostname
linux14
$ gcc -O3 -Wall -Wextra -Werror -pedantic -o gcd gcd.c
$ ./gcd
Usage: ./gcd <num1> <num2>
$ echo $?
1
$ ./gcd 1
```

```
Usage: ./gcd <num1> <num2>
$ ./gcd 18 42
gcd(18, 42) = 6
$ ./gcd 42 18
gcd(42, 18) = 6
$ ./gcd -42 18
gcd(-42, 18) = 6
$ ./gcd 982451653 776531401
gcd(982451653, 776531401) = 1
$ ./gcd 78423360000000000 35241600000000000
gcd(78423360000000000, 35241600000000000) = 960000000000
$ ./gcd 68719476736 84767329979727872
gcd(68719476736, 84767329979727872) = 68719476736
$ echo $?
0
$ time ./gcd 1000000000000000000 999999999999999999 gcd(1000000000000000000,
999999999999999999) = 1
real 0m0.009s
user 0m0.004s
sys 0m0.004s0
nan
```

Στο αρχείο README.md πρέπει να προσθέσετε οποιεσδήποτε παρατηρήσεις σας κατά την διεκπεραίωση της άσκησης. Ο κώδικας απαιτείται να είναι καλά τεκμηριωμένος με σχόλια καθώς αυτό θα είναι μέρος της βαθμολόγησης