Divisors - Λεκτική περιγραφή

Αναπτύξτε ένα πρόγραμμα σύμφωνα με το οποίο το σύστημα θα δέχεται ως είσοδο ένα αριθμό και θα ελέγχει αν αυτός είναι πρώτος (prime). Αν ΔΕΝ είναι πρώτος θα εμφανίζει ~~το πλήθος των αριθμών που τον διαιρούν ακριβώς~~ (τους διαιρέτες του).

Λεκτική Περιγραφή – Αφαιρετικότητα στις διεργασίες

- Πάρε τον αριθμό
- Αν ο **αριθμός είναι Πρώτος**
  - Ενημέρωσε τον χρήστη για αυτό
- Αλλιώς
  - **Εμφάνισε διαιρέτες του αριθμού**

Δώστε στην φάση αυτή μόνο την μετατροπή αυτής της ΛΠ σε πηγαίο κώδικα

Υπάρχουν ποικίλες μέθοδοι για να προσδιορίσουμε αν ένας αριθμός n είναι πρώτος. Η πιο βασική μέθοδος, η δοκιμαστική διαίρεση, έχει μικρή πρακτική χρησιμότητα επειδή είναι αργή. Ένα τμήμα των σύγχρονων μεθόδων για τον προσδιορισμό αν ένας αριθμός είναι πρώτος είναι εφαρμόσιμο για όλους τους αριθμούς, ενώ οι πιο αποτελεσματικές μέθοδοι είναι διαθέσιμες μόνο για συγκεκριμένες κατηγορίες αριθμών. Οι περισσότερες από αυτές τις μεθόδους λένε μόνο αν ο αριθμός είναι πρώτος ή όχι. Οι μέθοδοι, οι οποίες επιπλέον βρίσκουν και έναν ή περισσότερους παράγοντες του υπό εξέταση αριθμού ονομάζονται αλγόριθμοι παραγοντοποίησης. Πηγή: Wikipedia

Divisors – **ΛΠ ->Πηγαίος Κώδικας**

- Πάρε τον αριθμό
- Αν ο **αριθμός είναι Πρώτος**
  - Ενημέρωσε τον χρήστη για αυτό
- Αλλιώς
  - **Εμφάνισε διαιρέτες του αριθμού**

Function prototypes

- **int isPrime(int num);** //function prototype
- **void displayDivisors(int num);**
  //function prototype

- **Function declaration** ?
  - πρώτος αριθμός (ή απλά πρώτος) είναι ένας φυσικός αριθμός με την ιδιότητα οι μόνοι φυσικοί διαιρέτες του να είναι η μονάδα και ο εαυτός του.

[Δ2] Divisors – **Program structure**

```c
#include <stdio.h>
#include <stdlib.h>

int isPrime(int num);
void displayDivisors(int num);

/* run this program using the conso
int num;

int main(int argc, char *argv[]) {
    printf(...);
    scanf(....);
    if(isPrime(num))
        printf(....);
    else
        displayDivisors(num);
    return 0;
}

int isPrime(int num){

}

void displayDivisors(int num){

}
```

[Δ2] Divisors – **Αλγόριθμοι εύρεσης πρώτων**

Απλός 1 - από τον ορισμό του πρώτου αριθμού [Επεξεργασία | επεξεργασία κώδικα]

- Εξετάζουμε διαδοχικά όλους τους ακέραιους *M* < *N*
- Μόλις βρεθεί διαιρέτης του *N* σταματάμε και ο *N* δεν είναι πρώτος
- Αν εξαντληθούν οι *M* χωρίς να βρεθεί διαιρέτης, τότε ο *N* είναι πρώτος

Απλός 2 [Επεξεργασία | επεξεργασία κώδικα]

Βασιζόμενοι στην παρατήρηση ότι κανένας αριθμός N δεν έχει διαιρέτη μεγαλύτερο του N/2, τροποποιούμε τον παραπάνω αλγόριθμο εξετάζοντας όλους τους αριθμούς M < N/2, διπλασιάζοντας έτσι την ταχύτητα σε σχέση με τον "Απλό 1".

Απλός 3 [Επεξεργασία | επεξεργασία κώδικα]

Παρατηρούμε ότι αν ένας αριθμός *N* δεν είναι πρώτος τότε έχει (τουλάχιστον) δύο διαιρέτες μεγαλύτερους από *1*. Σε αυτήν την περίπτωση τουλάχιστον ένας διαιρέτης είναι μικρότερος από την τετραγωνική ρίζα του αριθμού. Τροποποιούμε τον αλγόριθμο 2 εξετάζοντας όλους τους αριθμούς M που είναι μικρότεροι από την τετραγωνική ρίζα του *N*, αν η τελευταία δεν είναι ακέραιος. Αλλιώς ο αριθμός *δεν* είναι πρώτος, επειδή τον διαιρεί και η τετραγωνική του ρίζα.

Πηγή: Wikipedia

[Δ3]AddNumbers – Incremental Development

Γράψτε ένα πρόγραμμα σύμφωνα με το οποίο η μηχανή θα υπολογίζει το άθροισμα ακέραιων αριθμών. Προσέξτε τους περιορισμούς!

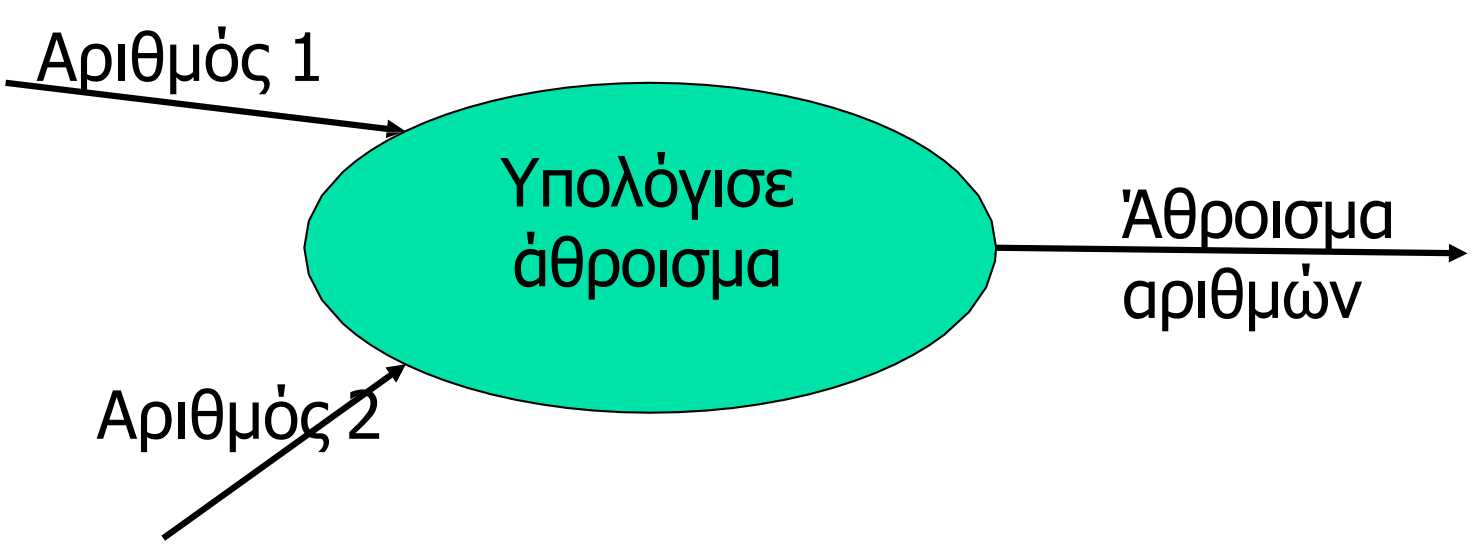
Περιορισμοί: Υποθέστε πως η μηχανή :

a) δεν μπορεί να υπολογίσει άθροισμα αριθμών με περισσότερα από ένα ψηφία. Για τον υπολογισμό του αθροίσματος 2 ψηφίων χρησιμοποιήστε τον τελεστή +.
b) μπορεί να κάνει τις πράξεις του πολλαπλασιασμού και διαίρεσης ακέραιων αριθμών με χρήση των τελεστών * και / αντίστοιχα. Επιπλέον μπορεί να βρίσκει το modulo ενός ακέραιου αριθμού με χρήση του τελεστή %.

Απαραίτητη η εφαρμογή της Αυξητικής Ανάπτυξης

- 1η έκδοση: Οι αριθμοί είναι 2 με 2 το πολύ ψηφία.
- 2η έκδοση: Οι αριθμοί είναι 2 με 3 το πολύ ψηφία.
- 3η έκδοση: Οι αριθμοί είναι 3 με 3 το πολύ ψηφία.
- 4η έκδοση: Οι αριθμοί είναι k με n το πολύ ψηφία**. Αξιοποιήστε προτάσεις επανάληψης και πίνακες.**

AddNumbers – Ενέργειες Ε1, Ε2




```
 3482
+ 635
-------
```

- Επόμενο Βήμα: Λεκτική περιγραφή
- Λεκτική περιγραφή είναι η περιγραφή, σε προστακτική μορφή, των ενεργειών που πρέπει να κάνει ο **μαθητής της Α δημοτικού** για να οδηγηθεί στο αποτέλεσμα με βάση τις εισόδους.

AddNumbers – Λεκτική περιγραφή

```
3482
+ 635
-------
```

- Βάλε τον μεγαλύτερο αριθμό επάνω.
- Βάλε από κάτω τον άλλο αριθμό με στοίχιση στα δεξιά
- Ξεκίνα από δεξιά και για κάθε ψηφίο του πάνω αριθμού
  - Πρόσθεσε το με το ψηφίο του κάτω αριθμού
  - Αν υπάρχει από προηγούμενη πρόσθεση κρατούμενο
    - πρόσθεσε το
  - Αν το άθροισμα είναι μικρότερο από 9
    - Γράψε το από κάτω
  - Αλλιώς
    - Γράψε από κάτω το δεξιό ψηφίο του αθροίσματος
    - Κράτα ως κρατούμενο το αριστερό ψηφίο του αθροίσματος
- Δώσε το αποτέλεσμα της άθροισης.

Προϋποθέτει:
Α) γνώση του αλγορίθμου, και
Β) δυνατότητα περιγραφής του στην ομιλούμενη γλώσσα
Ο πηγαίος κώδικας θα δημιουργηθεί στη συνέχεια με βάση την Λ.Π.