ΕΙΣΑΓΩΓΗ ΣΤΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ (2008-09)

Άσκηση 3

Μία *ακέραια διαμέριση* (integer partition) ενός θετικού ακεραίου N είναι η διάσπασή του σε θετικούς ακεραίους που το άθροισμά τους ισούται με N. Για παράδειγμα, μία ακέραια διαμέριση του 5 είναι οι αριθμοί 2, 2 και 1. Όλες οι δυνατές ακέραιες διαμερίσεις του 5 είναι οι εξής:

```
5
4 1
3 2
3 1 1
2 2 1
2 1 1 1
1 1 1 1 1
```

Παρατηρήστε ότι στις δυνατές ακέραιες διαμερίσεις του 5 δεν συμπεριλαμβάνεται και η `[2 1 2]`, αφού υπάρχει η `[2 2 1]`, και δεν έχει σημασία η σειρά με την οποία απαριθμούνται τα στοιχεία μίας ακέραιας διαμέρισης ενός αριθμού.

Έστω ότι επιθυμούμε να βρούμε πόσες είναι οι δυνατές ακέραιες διαμερίσεις ενός δεδομένου θετικού ακεραίου N. Ας συμβολίσουμε το πλήθος τους με $P(N)$. Μπορούμε να σκεφτούμε ως εξής για τον υπολογισμό του $P(N)$. Έστω $P_M(K, N)$ το πλήθος των ακέραιων διαμερίσεων του N οι οποίες αποτελούνται από αριθμούς **μεγαλύτερους ή ίσους** του K. Προφανώς ισχύει

$$P(N) = P_M(1, N) \tag{1}$$

Από τις $P_M(K, N)$ ακέραιες διαμερίσεις του N που αποτελούνται από αριθμούς μεγαλύτερους ή ίσους του K, κάποιες, έστω πλήθους $P_=(K, N)$, έχουν σαν ελάχιστο στοιχείο το K και κάποιες, έστω πλήθους $P_>(K, N)$, έχουν σαν ελάχιστο στοιχείο κάποιο που είναι γνησίως μεγαλύτερο του K. Προφανώς ισχύει:

$$P_M(K, N) = \begin{cases} 0, & \text{αν } K > N \\ 1, & \text{αν } K = N \\ P_=(K, N) + P_>(K, N), & \text{αν } K < N \end{cases} \tag{2}$$

Για παράδειγμα, για τις ακέραιες διαμερίσεις του 5, που δίνονται παραπάνω, ισχύουν τα εξής:

K	$P_M(K, 5)$	$P_=(K, 5)$	$P_>(K, 5)$
1	7	5	2
2	2	1	1
3	1	0	1
4	1	0	1
5	1	1	0

Οι $P_=(1, 5) = 5$ διαμερίσεις του 5 που έχουν ελάχιστο στοιχείο το 1 είναι οι `[4 1]`, `[3 1 1]`, `[2 2 1]`, `[2 1 1 1]` και `[1 1 1 1 1]`. Ενώ, οι $P_>(1, 5) = 2$ διαμερίσεις του 5 με ελάχιστο στοιχείο γνησίως μεγαλύτερο του 1 είναι οι `[5]` και `[3 2]`. Με παρόμοιο τρόπο, μπορείτε να εξηγήσετε και τις άλλες τιμές στον παραπάνω πίνακα.

Αναλύοντας λίγο περισσότερο το θέμα, μπορούμε να σκεφτούμε ότι αν από κάθε μία από τις $P_=(K, N)$ ακέραιες διαμερίσεις του N με ελάχιστο στοιχείο το K διαγράψουμε ένα K, θα πάρουμε

όλες τις ακέραιες διαμερίσεις του $N - K$ που αποτελούνται από αριθμούς μεγαλύτερους ή ίσους του K. Δηλαδή:

$$P_{=}(K, N) = P_M(K, N - K) \tag{3}$$

Επίσης, οι $P_{>}(K, N)$ ακέραιες διαμερίσεις του N με ελάχιστο στοιχείο γνησίως μεγαλύτερο του K είναι ακριβώς οι ίδιες με τις ακέραιες διαμερίσεις του N που αποτελούνται από αριθμούς μεγαλύτερους ή ίσους του $K + 1$. Δηλαδή:

$$P_{>}(K, N) = P_M(K + 1, N) \tag{4}$$

Οπότε, συνδυάζοντας τους τύπους (2), (3) και (4), παίρνουμε τον

$$P_M(K, N) = \begin{cases} 0, & \text{αν } K > N \\ 1, & \text{αν } K = N \\ P_M(K, N - K) + P_M(K + 1, N), & \text{αν } K < N \end{cases} \tag{5}$$

που σε συνδυασμό και με τον τύπο (1), μπορούμε πλέον να υπολογίσουμε το $P(N)$, το πλήθος των ακέραιων διαμερίσεων οποιουδήποτε θετικού ακεραίου N.

Αντικείμενο της άσκησης αυτής είναι να γράψετε δύο συναρτήσεις C, οι οποίες να υπολογίζουν το πλήθος των ακέραιων διαμερίσεων ενός θετικού ακεραίου N, τον οποίο θα παίρνουν σαν παράμετρο. Η μία συνάρτηση πρέπει να κάνει τον υπολογισμό με **αναδρομικό** τρόπο και η άλλη με **επαναληπτικό**. Και οι δύο συναρτήσεις θα πρέπει να επιστρέφουν στο όνομά τους το πλήθος των ακέραιων διαμερίσεων που υπολόγισαν. **Προαιρετικά**, μπορείτε να γράψετε και μία συνάρτηση C, η οποία να βρίσκει με **αναδρομικό** τρόπο και να εκτυπώνει όλες τις ακέραιες διαμερίσεις ενός θετικού ακεραίου N. Γράψτε επίσης και `main` συνάρτηση, η οποία θα παρέχει στον χρήστη ένα μενού για να επιλέγει ποια συνάρτηση από αυτές που έχετε γράψει θα καλέσει. Αν το εκτελέσιμο πρόγραμμα που θα κατασκευάσετε τελικά ονομάζεται "`intpart`", μία ενδεικτική εκτέλεσή του μπορεί να είναι η εξής:[1]

```
% ./intpart

Your options are:
1. Count integer partitions up to a number recursively
2. Count integer partitions up to a number iteratively
3. Generate integer partitions of a number recursively
4. Exit
Please, give your choice: 1

Please, enter a number N: 5
Number of integer partitions of   1 is: 1
Number of integer partitions of   2 is: 2
Number of integer partitions of   3 is: 3
Number of integer partitions of   4 is: 5
Number of integer partitions of   5 is: 7
```

[1] Έχετε απόλυτη ελευθερία για το πώς θα οργανώσετε το μενού επιλογής στη `main` συνάρτησή σας. Απλώς, πρέπει να επιδείξετε τη δουλειά σας με τις συναρτήσεις που υλοποιήσατε. Για παράδειγμα, δεν είναι απαραίτητο να έχετε επιλογή για τον υπολογισμό του πλήθους των ακέραιων διαμερίσεων για όλους τους ακεραίους μέχρι το δεδομένο N. Μπορείτε απλώς να υπολογίζετε το πλήθος των ακέραιων διαμερίσεων μόνο για το συγκεκριμένο N που δόθηκε. Επίσης, είναι αυτονόητο ότι αν δεν υλοποιήσετε την αναδρομική εύρεση όλων των ακέραιων διαμερίσεων ενός N, δεν θα υπάρχει και σχετική επιλογή στο μενού.

```
Your options are:
1. Count integer partitions up to a number recursively
2. Count integer partitions up to a number iteratively
3. Generate integer partitions of a number recursively
4. Exit
Please, give your choice: 2

Please, enter a number N: 5
Number of integer partitions of   1 is: 1
Number of integer partitions of   2 is: 2
Number of integer partitions of   3 is: 3
Number of integer partitions of   4 is: 5
Number of integer partitions of   5 is: 7

Your options are:
1. Count integer partitions up to a number recursively
2. Count integer partitions up to a number iteratively
3. Generate integer partitions of a number recursively
4. Exit
Please, give your choice: 1

Please, enter a number N: 70
Number of integer partitions of   1 is: 1
Number of integer partitions of   2 is: 2
Number of integer partitions of   3 is: 3
Number of integer partitions of   4 is: 5
Number of integer partitions of   5 is: 7
Number of integer partitions of   6 is: 11
Number of integer partitions of   7 is: 15
.....................................................
Number of integer partitions of  68 is: 3087735
Number of integer partitions of  69 is: 3554345
Number of integer partitions of  70 is: 4087968

Your options are:
1. Count integer partitions up to a number recursively
2. Count integer partitions up to a number iteratively
3. Generate integer partitions of a number recursively
4. Exit
Please, give your choice: 2

Please, enter a number N: 127
Number of integer partitions of   1 is: 1
Number of integer partitions of   2 is: 2
Number of integer partitions of   3 is: 3
Number of integer partitions of   4 is: 5
Number of integer partitions of   5 is: 7
Number of integer partitions of   6 is: 11
Number of integer partitions of   7 is: 15
```

```
Number of integer partitions of   8 is: 22
Number of integer partitions of   9 is: 30
Number of integer partitions of  10 is: 42
.....................................................
Number of integer partitions of 120 is: 1844349560
Number of integer partitions of 121 is: 2056148051
Number of integer partitions of 122 is: 2291320912
Number of integer partitions of 123 is: 2552338241
Number of integer partitions of 124 is: 2841940500
Number of integer partitions of 125 is: 3163127352
Number of integer partitions of 126 is: 3519222692
Number of integer partitions of 127 is: 3913864295

Your options are:
1. Count integer partitions up to a number recursively
2. Count integer partitions up to a number iteratively
3. Generate integer partitions of a number recursively
4. Exit
Please, give your choice: 3

Please, enter a number N: 5

Integer partitions of 5 are:
5
4 1
3 2
3 1 1
2 2 1
2 1 1 1
1 1 1 1 1

Your options are:
1. Count integer partitions up to a number recursively
2. Count integer partitions up to a number iteratively
3. Generate integer partitions of a number recursively
4. Exit
Please, give your choice: 3

Please, enter a number N: 11

Integer partitions of 11 are:
11
10 1
9 2
9 1 1
8 3
8 2 1
8 1 1 1
7 4
```

7 3 1
7 2 2
7 2 1 1
7 1 1 1 1
6 5
6 4 1
6 3 2
6 3 1 1
6 2 2 1
6 2 1 1 1
6 1 1 1 1 1
5 5 1
5 4 2
5 4 1 1
5 3 3
5 3 2 1
5 3 1 1 1
5 2 2 2
5 2 2 1 1
5 2 1 1 1 1
5 1 1 1 1 1 1
4 4 3
4 4 2 1
4 4 1 1 1
4 3 3 1
4 3 2 2
4 3 2 1 1
4 3 1 1 1 1
4 2 2 2 1
4 2 2 1 1 1
4 2 1 1 1 1 1
4 1 1 1 1 1 1 1
3 3 3 2
3 3 3 1 1
3 3 2 2 1
3 3 2 1 1 1
3 3 1 1 1 1 1
3 2 2 2 2
3 2 2 2 1 1
3 2 2 1 1 1 1
3 2 1 1 1 1 1 1
3 1 1 1 1 1 1 1 1
2 2 2 2 2 1
2 2 2 2 1 1 1
2 2 2 1 1 1 1 1
2 2 1 1 1 1 1 1 1
2 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1

```
Your options are:
1. Count integer partitions up to a number recursively
2. Count integer partitions up to a number iteratively
3. Generate integer partitions of a number recursively
4. Exit
Please, give your choice: 4
%
```

Μέχρι ποια τιμή του N η αναδρομική συνάρτηση που γράψατε ολοκληρώνει τον υπολογισμό της σε λογικό χρόνο; Μέχρι ποια τιμή του N η επαναληπτική συνάρτηση που γράψατε βρίσκει σωστό αποτέλεσμα; Γιατί για μεγαλύτερα N τα αποτελέσματα είναι λάθος; Απαντήστε σ΄ αυτές τις ερωτήσεις σε σχόλια μέσα στο πρόγραμμά σας.

Θα πρέπει να δομήσετε το πρόγραμμά σας σε ένα σύνολο από **τουλάχιστον δύο πηγαία αρχεία** C (με κατάληξη `.c`) και **τουλάχιστον ένα αρχείο επικεφαλίδας** (με κατάληξη `.h`).

Για να παραδώσετε το σύνολο των αρχείων που θα έχετε δημιουργήσει για την άσκηση αυτή, ακολουθήστε την εξής διαδικασία. Τοποθετήστε όλα τα αρχεία μέσα σ΄ ένα κατάλογο που θα δημιουργήσετε, έστω με όνομα `intpart`, στους σταθμούς εργασίας Suns ή Linux του Τμήματος. Επίσης, τοποθετήστε στον κατάλογο αυτό και ένα αρχείο με όνομα `README`, στο οποίο να δίνετε οδηγίες για τη μεταγλώττιση των αρχείων και την κατασκευή του τελικού εκτελέσιμου. Χρησιμοποιώντας την εντολή `zip` ως εξής

```
zip -r intpart.zip intpart
```

δημιουργείτε ένα συμπιεσμένο (σε μορφή `zip`) αρχείο, με όνομα `intpart.zip`, στο οποίο περιέχεται ο κατάλογος `intpart` μαζί με όλα τα περιεχόμενά του.[2] Το αρχείο αυτό είναι που θα πρέπει να υποβάλετε, με διαδικασία που θα ανακοινωθεί σύντομα.

[2] Αρχεία `zip` μπορείτε να δημιουργήσετε και στα Windows, με διάφορα προγράμματα, όπως, για παράδειγμα, το WinZip.