1η Άσκηση

Πρόβλημα 1 Θεωρήστε ένα σύνολο A ακεραίων αριθμών με πληθάριθμο n. Το σύνολο A θέλουμε να το *διαμερίσουμε* σε ένα σύνολο υποσυνόλων του $B_1, B_2, \ldots, B_k$ με τις παρακάτω ιδιότητες (οι τρεις πρώτες ιδιότητες αφορούν αποκλειστικά και μόνο το ότι το σύνολο των υποσυνόλων είναι διαμέριση του A):

1. Κάθε B_i είναι μη κενό.
2. Τα B_i είναι ανά δύο ξένα μεταξύ τους (δηλ. $B_i \cap B_j = \emptyset$, για κάθε $1 \leq i < j \leq k$).
3. Η ένωση όλων των B_i μας δίνει το A (δηλ. $\cup_{i=1}^{k} B_i = A$).
4. Δύο στοιχεία x και y ανήκουν στο ίδιο B_i αν και μόνο αν υπάρχουν $z_0, z_1, \ldots, z_{s+1} \in A$ τέτοια ώστε $z_0 = x$, $z_{s+1} = y$ και για κάθε ℓ, με $0 \leq \ell \leq s$, έχουμε $|z_{\ell+1} - z_\ell| \leq 1$.
5. Το μέγεθος k της διαμέρισης θα πρέπει να είναι το ελάχιστο δυνατό.

Είναι δυνατό (χωρίς αυτό να αποτελεί μέρος της εργασίας σας) να αποδειχθεί ότι η διαμέριση με τις παραπάνω ιδιότητες είναι μοναδική. Για παράδειγμα, αν $A = \{3, 4, 5, 3, 5, 2, 3, 7, 8, 9, 10, 12\}$ (οπότε $n = 12$), η διαμέριση που ψάχνουμε έχει μέγεθος $k = 3$, και ειδικότερα

$$B_1 = \{2, 3, 3, 3, 4, 5, 5\},$$
$$B_2 = \{7, 8, 9, 10\},$$
$$B_3 = \{12\}.$$

Σκοπός σας, στην άσκηση αυτή, είναι να γράψετε συνάρτηση για τον υπολογισμό της διαμέρισης με τις παραπάνω ιδιότητες. Θεωρήστε ότι σας δίνεται ένας πίνακας `a` ακεραίων αριθμών, με μέγεθος `n`. Σκοπός σας είναι να γράψετε μία συνάρτηση με όνομα `split_array` και δήλωση:

```
struct IntegerArray* split_array(int *a, unsigned int n, unsigned int* k);
```

η οποία στην θέση `*k` θα γράφει το μέγεθος της διαμέρισης, ενώ θα επιστρέφει ένα πίνακα από δομές `IntegerArray` (πίνακες ακεραίων), όπου κάθε πίνακας θα αντιστοιχεί σε σύνολο της διαμέρισης. Ο πίνακας που θα επιτρέφεται θα πρέπει να δεσμεύεται δυναμικά, και να έχει πλήθος στοιχείων ακριβώς `*k`. Η δομή `IntegerArray` θα πρέπει να έχει δύο πεδία `elements` και `size`. Το πεδίο `elements` θα πρέπει να είναι δείκτης σε ακεραίους που θα αντιστοιχεί στον πίνακα με ακεραίους και θα δεσμεύεται δυναμικά, το δε πεδίο `size` θα είναι μη προσημασμένος ακέραιος με τιμή ίση με το πλήθος των στοιχείων του `elements`. Τα στοιχεία του πίνακα `elements` θα πρέπει να είναι καταχωρημένα σε αύξουσα σειρά και το πλήθος των στοιχείων του να είναι ακριβώς `size`.

Η συνάρτησή σας θα πρέπει να επιστρέφει `NULL` αν η τιμή του `n` είναι 0, όπως επίσης και σε κάθε περίπτωση που η δυναμική δέσμευση μνήμης αποτύχει.

Αναφερόμενοι στο παράδειγμα πιο πάνω, αν `a={3,4,5,3,5,2,3,8,7,9,12,10}`, η συνάρτηση θα πρέπει να επιστρέφει και στην επιστροφή η τιμή του `*k` θα πρέπει να είναι `3`, ενώ αν ονομάσουμε `b` τον δείκτη σε `IntegerArray` που επιστρέφεται, θα πρέπει να έχουμε:

Θέση πίνακα	Τιμή
`b[0].elements[0]`	2
`b[0].elements[1]`	3
`b[0].elements[2]`	3
`b[0].elements[3]`	3
`b[0].elements[4]`	4
`b[0].elements[5]`	5
`b[0].elements[6]`	5

Θέση πίνακα	Τιμή
`b[1].elements[0]`	7
`b[1].elements[1]`	8
`b[1].elements[2]`	9
`b[1].elements[3]`	10
`b[2].elements[0]`	12
`b[0].size`	7
`b[1].size`	4
`b[2].size`	1

Για διευκόλυνσή σας στην ιστοσελίδα του μαθήματος θα βρείτε το αρχείο `sort_int_array.h` στο οποίο ορίζεται η συνάρτηση

```
void sort_int_array(int* arr, unsigned int n);
```

η οποία ταξινομεί τον πίνακα ακεραίων αριθμών `arr` μεγέθους `n` σε αύξουσα σειρά. Το αρχείο αυτό μπορείτε να το χρησιμοποιήσετε σε πρόγραμμά σας σώζοντάς το στον ίδιο κατάλογο με το αρχείο του προγράμματός σας και γράφοντας την εντολή

```
#include "sort_int_array.h"
```

στην αρχή του.

Επίσης για διευκόλυνσή σας, στην ιστοσελίδα του μαθήματος θα βρείτε το αρχείο `split_array.h` στο οποίο μπορείτε να συμπληρώσετε τον κώδικά σας. Το αρχείο αυτό, στη συνέχεια, μπορείτε να το χρησιμοποιήσετε σε πρόγραμμά σας σώζοντάς το στον ίδιο κατάλογο με το αρχείο του προγράμματός σας και γράφοντας την εντολή

```
#include "split_array.h"
```

στην αρχή του.