Γενικά σχόλια lab9

Μεταγλώττιση: Ο κώδικας θα πρέπει να μεταγλωττίζεται χωρίς **warnings**. Χαρακτηρισμός "μη ικανοποιητικό" σημαίνει ότι είχατε τουλάχιστον ένα warning. Κώδικας που παράγει λάθη (errors) κατά τη μεταγλώττιση βαθμολογείται με FAIL.

Στοίχιση: Ο κώδικας πρέπει να είναι στοιχισμένος σωστά, με ιδιαίτερη προσοχή στη στοίχιση εμφωλευμένων for/while/if. Χαρακτηρισμός "μέτρια" γενικά σημαίνει ότι έχετε κατά το πλείστον καλή στοίχιση Χαρακτηρισμός "μη ικανοποιητική" σημαίνει ότι δεν έχετε στοιχίσει τον κώδικά σας ή η στοίχιση είναι πολύ ασυνεπής ειδικά όσον αφορά τις δομές ελέγχου/επανάληψης.

Ονόματα: Όπως πάντα, θέλουμε περιγραφικά ονόματα συναρτήσεων και μεταβλητών. Για παράδειγμα, το όνομα function ή sinartisi για μία συνάρτηση δεν είναι καλό γιατί δε μας λέει τι κάνει η συγκεκριμένη συνάρτηση. Ανάλογα τα ονόματα a, b, m, x, y, z, variable, metabliti, array κλπ δεν είναι καλά ονόματα μεταβλητών διότι δεν προσδιορίζουν το σκοπό για τον οποίο χρησιμοποιείται η συγκεκριμένη μεταβλητή. Συνιστάται να διαλέγουμε ρηματικές φράσεις ως ονόματα συναρτήσεων διότι κάθε συνάρτηση κάνει κάποια ενέργεια. Για παράδειγμα get_limit και όχι απλά limit.
Ονόματα τοπικών μεταβλητών διαφορετικών συναρτήσεων μπορούν να είναι ίδια, και ενδείκνυται κάτι τέτοιο αν πρόκειται για όμοιες ποσότητες. Ονόματα του ενός γράμματος πρέπει να αποφεύγονται με μόνη εξαίρεση μετρητές για for ή while loops.

Χρήση sprintf: Πέρα από το αν είναι σωστή η χρήση sprintf για τη δημιουργία του format string στην άσκηση 1, ελέγχουμε αν είναι σωστά ορισμένο το μέγεθος του string κατά τη δήλωση: έχει μέγεθος 16 χωρίς το \0 άρα θα πρέπει να έχετε είτε #define SIZE 17, δήλωση συμβολοσειράς char str[SIZE] και χρήση SIZE-1 στην sprintf είτε #define SIZE 16, δήλωση συμβολοσειράς char str[SIZE+1] και χρήση SIZE στην sprintf. Επιπλέον, το μέγεθος του format string δεν έχει καμία σχέση με το SIZE. Πρέπει να είναι τόσο μεγάλο όσο χρειάζεται για να χωρέσει μέσα σε αυτό ο χαρακτήρας % , ένας ακέραιος και το γράμμα s.

Ορισμός και κλήσεις συναρτήσεων:

Άσκηση 1η: Στην 1η άσκηση καλείστε να γράψετε μία συνάρτηση η οποία λαμβάνει ως παραμέτρους ένα μονοδιάστατο πίνακα χαρακτήρων και δύο δείκτες σε χαρακτήρανα προκειμένου να διαβάσετε μία συμβολοσειρά και δύο χαρακτήρες. Σκοπός της συνάρτησης είναι να επιστρέψετε την πληροφορία που διαβάσατε μέσα από τις διευθύνσεις που περάσατε ως παραμέτρους. Το prototype της συνάρτησης θα μπορούσε να είναι το παρακάτω

```
void read_info(char str[], char *find_char, char *replace_char)
```

και η κλήση της θα μπορούσε να είναι η εξής

```
char str[SIZE+1], find, replace;
read_info(str, &find, &replace);
```

Διάβασμα της τιμής ενός χαρακτήρα ή ενός αλφαριθμητικού μέσα σε συνάρτηση και αποθήκευση σε διεύθυνση που δίνεται ως παράμετρος (όρισμα) στη συνάρτηση.

Ας υποθέσουμε ότι θέλουμε να διαβάσουμε έναν χαρακτήρα μέσα στη συνάρτηση

```
void read_char(char *char_ptr)
```

Η συνάρτηση θα μπορούσε να γραφεί με τους εξής παρακάτω δύο τρόπους:

`void read_char(char *char_ptr) {`
`  scanf(" %c", char_ptr);`
`}`
	`void read_char(char *char_ptr) {`
	`  char tmpc;`
	`  scanf(" %c", &tmpc);`
	`  *char_ptr = tmpc;`
	`}`

Διάβασμα και αποθήκευση του χαρακτήρα στη διεύθυνση που δείχνει ο δείκτης **char_ptr**.	Διάβασμα σε τοπική μεταβλητή και αποθήκευση του περιεχομένου της τοπικής μεταβλητής στο περιεχόμενο της διεύθυνσης που δείχνει ο δείκτης **char_ptr**.

Αντίστοιχα για την ανάγνωση και επιστροφή της πληροφορίας ενός αλφαριθμητικού η ανάγνωση και η κλήση της συνάρτησης θα μπορούσε να γίνει όπως παρακάτω:

Συνάρτηση ανάγνωσης αλφαριθμητικού	Κλήση της συνάρτησης ανάγνωσης αλφαριθμητικού.

`void read_str(char str[]) {`
`  char format_str[13];`
`  sprintf(format_str, "%%%ds", SIZE);`
`  scanf(format_str, str);`
`}`
	`char str[SIZE+1];`
	`read_str(str);`

Στη 2η συνάρτηση που ζητείται να γίνει αντικατάσταση όλων των εμφανίσεων ενός χαρακτήρα με άλλο στη συμβολοσειρά είναι απαραίτητο να περάσετε ως όρισμα τη διεύθυνση της συμβολοσειράς, και τις τιμές των δύο χαρακτήρων. Η συνάρτηση επιστρέφει τον αριθμό των αντικαταστάσεων και το prototype της είναι:

```
int replace( char str[], char find_char, char replace_char)
```

Η κλήση της (με βάση τους προηγούμενους ορισμούς μεταβλητών) είναι:

```
int replacements;
replacements = replace(str, find, replace);
```

Άσκηση 2η: Σε αυτή τη άσκηση καλείστε να αρχικοποιήσετε ένα πίνακα ακεραίων μεγέθους SIZE

```
int grades[SIZE]; // πρέπει να αρχικοποιήσετε τις τιμές του
                  // εκτός του εύρους τιμών [0,10].
```

Στη συνέχεια, καλείστε να γράψετε μία συνάρτηση η οποία λαμβάνει ως όρισμα τη διύεθυνση ενός ακέραιου, διαβάζει μία τιμή ακεραίου από το πληκτρολόγιο και την επιστρέφει. Μέσω της συνάρτησης αυτής θέλουμε να γεμίσουμε τον πίνακα ακεραίων. Το prototype της συνάρτησης δίνεται παρακάτω και η διαδικασία διαβάσματος μέσω της scanf είναι απολύτως ανάλογη με την διαδικασία διαβάσματος ενός χαρακτήρα που δόθηκε παραπάνω.

```
void read_grade(int *grade)
```

Για να διαβάσουμε και να αποθηκεύσουμε μία τιμή στη θέση i (I μεταξύ 0 και SIZE-1) του πίνακα **grades**, αρκεί να την καλέσουμε ως εξής:

```
read_grade( &grades[i] )
```

ή

```
read_grade(grades+i)
```

Ο υπολογισμός του μέσου όρου γίνεται από τη συνάρτηση υπολογισμού του μέσου όρου της οποίας το prototype είναι

```
void average(int grades[], int size, double *avg_ptr)
```

Στη διεύθυνση **avg_ptr** αποθηκεύεται η επιστρεφόμενη τιμή του μέσου όρου που υπολογίστηκε.

Άσκηση 3η: Το prototype της read_info γίνεται

```
void read_info(char *str, char *find_char, char *replace_char)
```

η υλοποίηση της read_info δεν αλλάζει. Το prototype της replace γίνεται

```
int replace( char *str, char find_char, char replace_char)
```

η υλοποίηση της αλλάζει ώστε αντί να χρησιμοποιείται την έκφραση **str[i]** για να προσπελάσετε ένα χαρακτήρα της συμβολοσειράς χρησιμοποιείτε το ισοδύναμο περιεχόμενο της διεύθυνσης ***(str+i)** .

Η συνάρτηση **replace** θα μπορούσε να γραφεί με τους παρακάτω δύο τρόπους.

```
void replace( char* str,
              char find,
              char replace) {
  while(*str != '\0') {
    if( *str == find )
      *str = replace;
    str++;
  }
}
```

```
void replace( char* str,
              char find,
              char replace) {
  int i=0;
  while( *(str+i) != '\0') {
    if( *(str+i) == find )
      *(str+i) = replace;
    i++;
  }
}
```

Άσκηση 4η:

Ο υπολογισμός του μέσου όρου με χρήση δεικτών γίνεται σε αναλογία με την άσκηση 3.

Ορθότητα αποτελεσμάτων: Ελέγχουμε και στις δύο ασκήσεις εάν διαβάσαμε σωστά τα δεδομένα εισόδου για κάθε άσκηση και εάν τα δεδομένα αυτά διατηρούνται και μετά την επιστροφή στο κυρίως πρόγραμμα από τη συνάρτηση ανάγνωσης των δεδομένων.

Στην 1η άσκηση ελέγχουμε ότι το αλφαριθμητικό μεταβάλλεται σωστά μετά την επιστροφή από τη συνάρτηση αντικατάστασης.

Στην 2η άσκηση ελέγχουμε ότι έχουν γίνει σωστά οι υπολογισμοί, λαμβάνοντας υπόψη το σύνολο των τιμών που διαβάστηκαν από το πληκτρολόγιο.

Ελέγξτε τα αποτελέσματα σας με τη βοήθεια των αρχείων ελέγχου που παρέχονται.

Έξοδος προγράμματος: Η έξοδος του προγράμματος πρέπει να είναι ίδια με την εκφώνηση.