ΕΡΓΑΣΤΗΡΙΟ 1

Θέμα: Μορφοποίηση κώδικα, Είσοδος/Εξοδος, Αριθμητικές πράξεις

Μορφοποίηση κώδικα

Οι περισσότερες γλώσσες προγραμματισμού παρέχουν ευελιξία στη διάταξη του κώδικα κι έχουν λίγους περιορισμούς για την ονομασία συναρτήσεων και μεταβλητών. Οι ελάχιστες απαιτήσεις είναι το πρόγραμμα να μεταγλωττίζεται χωρίς λάθη και να λειτουργεί σωστά σύμφωνα με τις δεδομένες προδιαγραφές.

Αυτό που πολλοί αρχάριοι προγραμματιστές ξεχνούν είναι πως είναι εξίσου σημαντικό το πρόγραμμα να μπορεί να συντηρηθεί (maintainable code), δηλαδή να είναι εύκολο να γίνουν διορθώσεις, βελτιώσεις, και προσθήκες νέων λειτουργιών σε βάθος χρόνου.

Υπάρχουν πολλά στοιχεία σε ένα πρόγραμμα τα οποία συντελούν στη συντηρησιμότητα του. Σε κάθε εργασία κι εργαστηριακή άσκηση, θα παρουσιάζουμε κάποια από αυτά τα στοιχεία. Παράλληλα, θα προσδιορίζουμε τους κανόνες που πρέπει να ακολουθούν όλα τα προγράμματα που γράφετε στα πλαίσια του μαθήματος.

Σε αυτό το εργαστήριο θα μιλήσουμε για στοιχεία διάταξης κώδικα (τοποθέτηση αγκίστρων, στοίχιση, κενά, κενές γραμμές) και ονομασία μεταβλητών.

Τοποθέτηση αγκίστρων

Συναρτήσεις όπως η `main` καθώς και σύνθετες εντολές που θα μάθετε αργότερα χρησιμοποιούν άγκιστρα για να οριοθετούν το "σώμα" τους, δηλαδή τη συλλογή εντολών που συνδέονται άμεσα με αυτές.

Οι δύο πιο δημοφιλείς τρόποι τοποθέτησης αγκίστρων είναι:

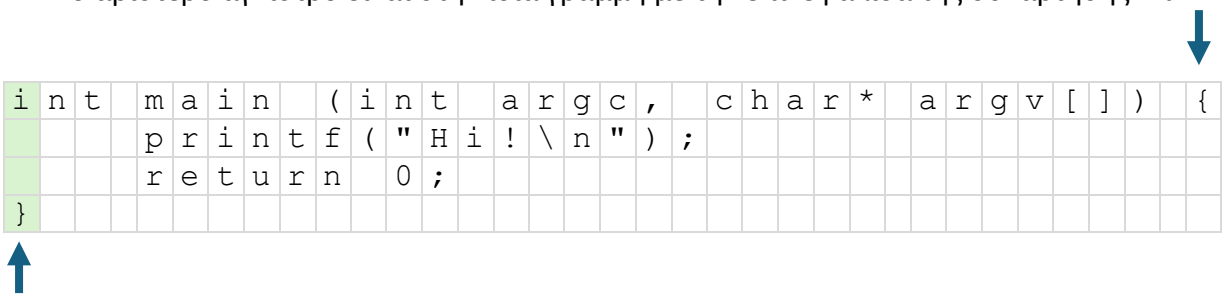
- Το αριστερό άγκιστρο βρίσκεται στην ίδια γραμμή με την επικεφαλίδα της συνάρτησης ή σύνθετη εντολή. Το δεξί άγκιστρο βρίσκεται στην ίδια στήλη με την αρχή της επικεφαλίδας της συνάρτησης ή σύνθετης εντολής, σε ξεχωριστή γραμμή που δεν περιλαμβάνει τίποτα άλλο.

- Το αριστερό και δεξί άγκιστρο βρίσκονται σε ξεχωριστές γραμμές, μόνα τους, στην ίδια στήλη με την αρχή της επικεφαλίδας της συνάρτησης ή σύνθετης εντολής.

Επιλέξτε όποιον από τους δύο τρόπους προτιμάτε, και μείνετε συνεπείς σε αυτόν.

Παράδειγμα 1ου τρόπου τοποθέτησης { }

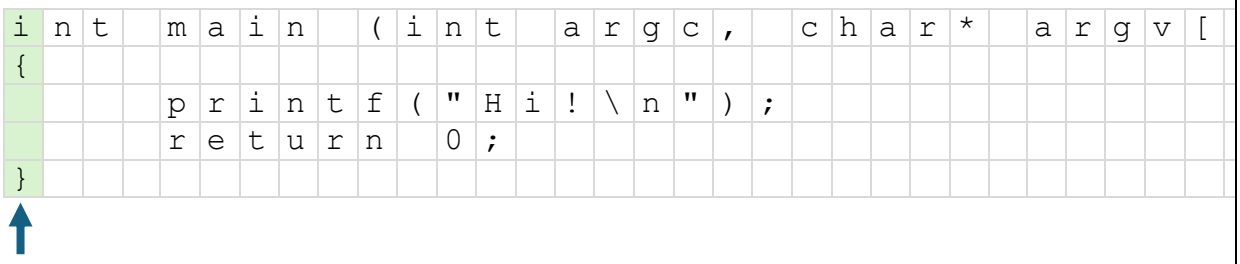
Το αριστερό άγκιστρο είναι στην ίδια γραμμή με την επικεφαλίδα της συνάρτησης main.



```
int main (int argc, char* argv[]) {
    printf("Hi!\n");
    return 0;
}
```

Το δεξί άγκιστρο είναι στην ίδια στήλη με την αρχή της επικεφαλίδας της συνάρτησης main.

Παράδειγμα 2ου τρόπου τοποθέτησης { }



```
int main (int argc, char* argv[]
{
    printf("Hi!\n");
    return 0;
}
```

Το αριστερό και το δεξί άγκιστρο είναι στην ίδια στήλη με την αρχή της επικεφαλίδας της συνάρτησης main, κάθε ένα σε νέα γραμμή.

Στοίχιση

Κάθε εντολή πρέπει να βρίσκεται σε ξεχωριστή γραμμή.

Εντολές που περικλείονται σε άγκιστρα { } (προς το παρόν όλες οι εντολές εντός της `main`) πρέπει να γράφονται 4 κενά πιο δεξιά από την εντολή/συνάρτηση στην οποία υπάγονται.

Δείτε στο παραπάνω παράδειγμα ότι οι εντολές `printf` και `return` ξεκινούν 4 κενά πιο δεξιά από τη στήλη που ξεκινά η συνάρτηση `main`.

Κενά

Αφήνετε ένα κενό πριν το αριστερό άγκιστρο που οριοθετεί το σώμα μια συνάρτησης ή σύνθετης εντολής αν χρησιμοποιείτε τον πρώτο τρόπο τοποθέτησης αγκίστρων.

Αφήνετε ένα κενό μετά από κάθε κόμμα (π.χ. σε δηλώσεις πολλαπλών μεταβλητών ίδιου τύπου).

Κενές γραμμές

Αφήνετε μια κενή γραμμή ανάμεσα σε διαφορετικά νοητά "τμήματα" εντός της ίδιας συνάρτησης. Για παράδειγμα, ανάμεσα στο τμήμα δήλωσης μεταβλητών και στο τμήμα εντολών, ή ανάμεσα στο κομμάτι που διαβάζει τα δεδομένα και στο κομμάτι που τα επεξεργάζεται.

Μην υπερβάλλετε. Μια κενή γραμμή κάθε μερικές γραμμές κώδικα είναι καλή. Πολλές διαδοχικές αλλοιώνουν τη ροή του κώδικα, αναγκάζουν το μάτι να "ταξιδεύει" πολύ στην οθόνη και μειώνουν αισθητά την ποσότητα κώδικα που χωρά στην οθόνη. Εξίσου άσχημο είναι να υπάρχει μια κενή γραμμή μετά από κάθε μία γραμμή κώδικα.

Αποφεύγετε να έχετε γραμμές μακρύτερες από 80 χαρακτήρες γιατί είναι δύσκολο να διαβαστούν. Αν μια έκφραση είναι πολύ μεγάλη, προσπαθήστε να τη χωρίσετε σε δύο εκφράσεις. Αργότερα θα σας δείξουμε πώς μπορείτε να συνεχίσετε μία εντολή στην επόμενη γραμμή.

Ονόματα μεταβλητών

Το όνομα μιας μεταβλητής πρέπει να είναι **συνοπτικό** και να **περιγράφει** πλήρως την οντότητα που εκπροσωπεί η μεταβλητή. Αποφεύγετε ονόματα με περισσότερα από 15 γράμματα. Τα ονόματα μεταβλητών γράφονται πάντα με μικρά εκτός από μια εξαίρεση που αναφέρεται παρακάτω.

Αν χρειαστεί, μπορείτε να χρησιμοποιήσετε σύνθετο όνομα, δηλαδή όνομα που αποτελείται από δύο λέξεις. Υπάρχουν δύο τρόποι γραφής σύνθετων ονομάτων:

- Το πρώτο γράμμα της δεύτερης λέξης γράφεται με κεφαλαίο και όλα τα άλλα με μικρά, π.χ. `salesTax`
- Τα γράμματα είναι όλα με μικρά και οι λέξεις διαχωρίζονται με μία κάτω παύλα, π.χ. `sales_tax`

Επιλέξτε όποιον από τους δύο τρόπους προτιμάτε, και μείνετε συνεπείς σε αυτόν.

Αποφεύγετε συντομεύσεις. Αν είναι απολύτως απαραίτητο να συντομεύσετε ένα όνομα φροντίστε να το κάνετε με τρόπο που διατηρεί τη σημασία του. Για παράδειγμα, το `studentName` δεν είναι καλό να συντομευθεί σε `studName` γιατί το stud έχει διαφορετική σημασία. Το `price_including_sales_tax` πρέπει σίγουρα να συντομευθεί γιατί είναι μεγάλο, αλλά δεν είναι καλό να συντομευθεί σε `pr_inc_slsTx` γιατί χάνεται η σημασία του. Μια καλύτερη συντόμευση θα ήταν `price_incl_tax` ή `price_w_tax`.

Αποφεύγετε παρόμοια ονόματα. Για παράδειγμα, είναι κακή ιδέα να βρίσκονται στο ίδιο πρόγραμμα δύο μεταβλητές με ονόματα `totalPrice` και `finalPrice` γιατί δεν είναι σαφής η διαφορά ανάμεσα τους.

Εκτός από πολύ συγκεκριμένες εξαιρέσεις που θα αναφέρουμε αργότερα, μη χρησιμοποιείτε **ποτέ ονόματα του ενός ή δύο γραμμάτων**.

Σχόλια

Ένα πρόγραμμα πρέπει να έχει πάντα ένα σχόλιο στην αρχή που να περιγράφει συνοπτικά τι κάνει το πρόγραμμα.

Απαιτήσεις

Τα προγράμματα σας πρέπει:

- να ακολουθούν πάντα τους κανόνες μορφοποίησης,
- να ακολουθούν πάντα τους κανόνες ονομασίας μεταβλητών,
- να ακολουθούν πάντα τους κανόνες σχολιασμού,
- να έχουν σωστούς τύπους,
- να παράγουν έξοδο που συμμορφώνεται πλήρως με τις προδιαγραφές.

Διευκρινίσεις

Τα τμήματα μηνυμάτων που εμφανίζονται με μαύρο χρώμα αποτελούν "σταθερό" κείμενο και πρέπει να εμφανίζονται ή να εισάγονται ακριβώς όπως δίνονται. Τα τμήματα που εμφανίζονται με κόκκινο χρώμα αντιστοιχούν σε τιμές ή αναφορές μεταβλητών.

Παράδειγμα 1:
Ζητούμενο: Διαβάστε ένα μήνυμα της μορφής `Year: E` όπου `E` το έτος.
Γράφετε: `scanf("Year: %d", &year);`
Παράδειγμα 2:
Ζητούμενο: Εκτυπώσετε το μήνυμα "`Year = E` " όπου `E` το έτος.
Γράφετε: `printf("Year = %d", year);`

Σημείωση: Και στα δύο παραδείγματα θεωρούμε δεδομένο πως έχει ήδη δηλωθεί μια ακέραια μεταβλητή με όνομα `year` για τις ανάγκες του υποθετικού προγράμματος.

Άσκηση 1

Το πρόγραμμα σας για την άσκηση 1 πρέπει να αποθηκευτεί σε αρχείο με όνομα **`lab1a.c`**

Μια καντίνα πουλάει μόνο καφέδες και σάντουιτς. Ο καφές χρεώνεται 1.70 ευρώ χωρίς ΦΠΑ και το σάντουιτς 3.50 ευρώ χωρίς ΦΠΑ. Το ποσοστό ΦΠΑ για τους καφέδες είναι 13% και για τα σάντουιτς 24%.

Γράψτε ένα πρόγραμμα που κάνει τα εξής:

1. Εκτυπώνει το μήνυμα "`Order:\n`"
2. Διαβάζει από το πληκτρολόγιο μια παραγγελία στη μορφή `XC YS` όπου `X` ένας ακέραιος που αναπαριστά πλήθος καφέδων και `Y` ένας ακέραιος που αναπαριστά πλήθος σάντουιτς. Τα `C`, `S` εμφανίζονται ως έχουν για να δηλώσουν καφέ και sandwich αντίστοιχα. Για παράδειγμα, για μια παραγγελία τεσσάρων καφέδων και δύο σάντουιτς, θα εισαχθεί `4C 2S`
3. Υπολογίζει το ΦΠΑ για κάθε είδος καθώς και όλες τις άλλες επιμέρους ποσότητες που θα εμφανιστούν στην απόδειξη (δείτε βήμα 4).
4. Εκτυπώνει μια απόδειξη που μοιάζει ως εξής:

```
Coffee: Tc
VAT: Vc
Sandwich: Ts
VAT: Vs
Subtotal: S
Total VAT: V
Total: T
```

όπου:

- `Tc` η τιμή όλων των καφέδων χωρίς ΦΠΑ,
- `Vc` ο συνολικός ΦΠΑ για τους καφέδες,
- `Ts`, `Vs` αντίστοιχα για τα σάντουιτς,
- `S` το άθροισμα των `Tc`, `Ts`
- `V` το άθροισμα των `Vc`, `Vs`
- `T` το άθροισμα των `S`, `V`

Όλα τα ποσά εκτυπώνονται με 2 δεκαδικά ψηφία. Στο τέλος κάθε γραμμής της απόδειξης βρίσκεται ένας χαρακτήρας αλλαγής γραμμής (`'\n'`).

Για 4 καφέδες και 3 σάντουιτς (δηλαδή για είσοδο `4C 3S`), η απόδειξη που θα εκτυπωθεί πρέπει να μοιάζει ως εξής:

```
Order:
Coffee: 6.80
VAT: 0.88
Sandwich: 10.50
VAT: 2.52
Subtotal: 17.30
Total VAT: 3.40
Total: 20.70
```

Άσκηση 2

Το πρόγραμμα σας για την άσκηση 2 πρέπει να αποθηκευτεί σε αρχείο με όνομα **`lab1b.c`**

Είστε κυβερνήτης ενός διαστημικού σταθμού στον Άρη που πρέπει να εκκενωθεί άμεσα. Οι κάψουλες διαφυγής (escape pods) χωρούν το πολύ 12 άτομα η κάθε μία. Στόχος σας είναι να γεμίσουν όσο το δυνατό περισσότερες κάψουλες και, αν υπάρχουν ακόμη μερικά άτομα, να μπουν σε μια τελευταία κάψουλα διαφυγής.

Μια εντελώς γεμάτη κάψουλα διαφυγής απαιτεί ακριβώς 150.00 gigajoules ενέργειας για να εκτοξευθεί. Αν η κάψουλα διαφυγής είναι μερικώς γεμάτη, τότε η ενέργεια που απαιτείται είναι ανάλογη του αριθμού επιβατών (π.χ. αν έχει 6 άτομα, απαιτείται η μισή ενέργεια).

Γράψτε ένα πρόγραμμα που κάνει τα εξής:

1. Εκτυπώνει το μήνυμα "`How many people?\n`"
2. Διαβάζει από το πληκτρολόγιο το συνολικό πλήθος ατόμων που βρίσκονται στο διαστημικό σταθμό.
3. Υπολογίζει πόσες εντελώς γεμάτες κάψουλες διαφυγής θα εκτοξευθούν.
4. Υπολογίζει πόσα άτομα περισσεύουν ώστε να επιβιβαστούν στην τελευταία κάψουλα διαφυγής.
5. Υπολογίζει τη συνολική ενέργεια που απαιτείται για την εκτόξευση.
6. Εκτυπώνει το μήνυμα:

```
Full pods: F
Remaining people: R
Energy: E
```

όπου:

- `F` το πλήθος γεμάτων καψουλών διαφυγής,
- `R` το πλήθος ατόμων που περισσεύουν,
- `E` η συνολική ενέργεια εκτόξευσης με 1 δεκαδικό ψηφίο

Στο τέλος κάθε γραμμής του μηνύματος βρίσκεται ένας χαρακτήρας αλλαγής γραμμής (`'\n'`).

Υποβολή

1. Ακολουθήστε τις οδηγίες στο φυλλάδιο του autolab για να σχηματίσετε ομάδα.
2. Η υποβολή θα γίνει από το ένα μέλος της ομάδας (δεν έχει σημασία ποιο).
3. Βεβαιωθείτε ότι έχετε δώσει σωστά ονόματα στα αρχεία .c που θα υποβάλετε
4. Επειδή το lab1 αποτελείται από δύο αρχεία, πρέπει να τα "πακετάρετε" πριν τα υποβάλετε. Αυτό μπορείτε να το κάνετε είτε από το τερματικό είτε μέσω γραφικού περιβάλλοντος.
    - Μέσω τερματικού: Μεταβείτε με χρήση cd στον κατάλογο που βρίσκονται τα αρχεία .c και γράψτε: `zip lab1.zip lab1a.c lab1b.c`
    - Ή, μέσω γραφικού περιβάλλοντος: Μεταβείτε με χρήση Dolphin στον κατάλογο που βρίσκονται τα αρχεία .c, επιλέξτε τα (μόνο αυτά) κάντε δεξί κλικ, κι επιλέξτε από το μενού Compress και μετά Here (as ZIP).
5. Σε κάθε περίπτωση θα πρέπει να έχει δημιουργηθεί το αρχείο `lab1.zip` το οποίο υποβάλει στο autolab το ένα μέλος της ομάδας.