Εισαγωγή στην Πληροφορική & στον Προγραμματισμό

Αρχές Προγραμματισμού Η/Υ (με τη γλώσσα C)

Γενική Ανακεφαλαίωση

6 & 7 Ιουνίου 2024

Παναγιώτης Παύλου

c-programming-24@allos.gr

Ανακεφαλαίωση Α’ μέρος

Γενική επανάληψη

Εφαρμογή A1 – Ένα… άτομο

Δημιουργήστε έναν νέο τύπο δεδομένων `Person` που να περιγράφει ένα άτομο με τα εξής στοιχεία:

- Όνομα
- Επώνυμο
- Ύψος σε εκατοστά
- Βάρος σε κιλά

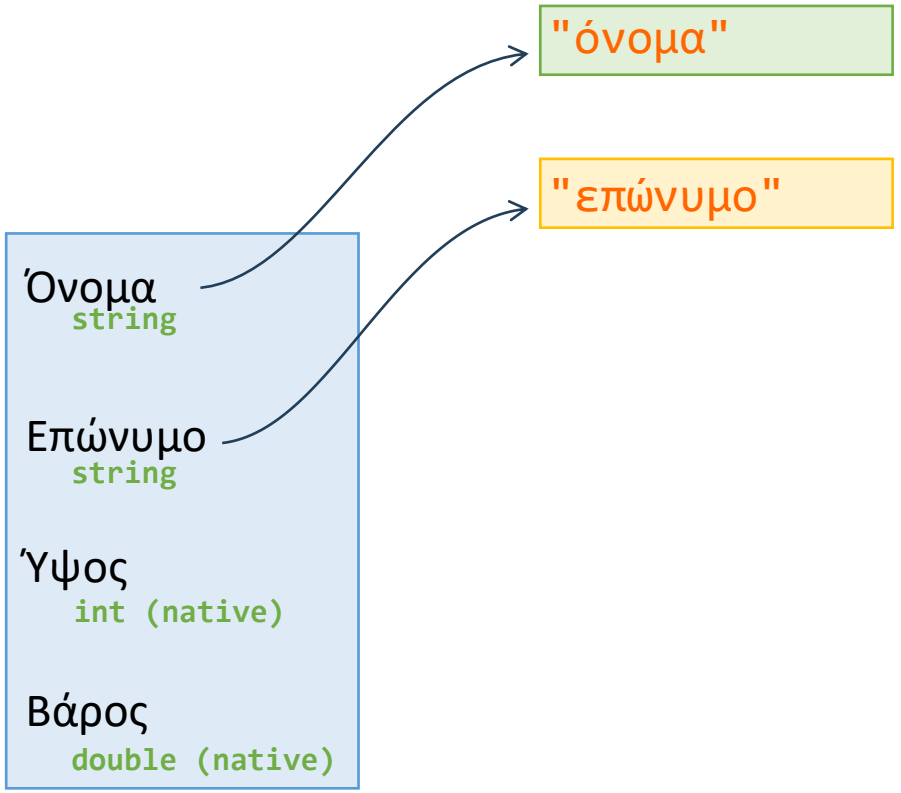
Για τη δομή αυτή δημιουργήστε έναν constructor:

```
Person *prsnCreate(char *first, char *last, int height, double weight)
```

και έναν destructor:

```
void prsnDelete(Person *)
```

Εφαρμογή Α1 – Ένα… άτομο



Εφαρμογή Α2 – Μορφοποίηση

Στους παρακάτω κώδικες τι εμφανίζουν στην οθόνη οι **printf** ? Αιτιολογείστε τις απαντήσεις σας.

```
int main(void) {
    unsigned char a = 0xB4;

    printf("X : %x\n", a);
    printf("O : %o\n", a);
    printf("D : %d\n", a);

    return 0;
}
```

```
int main(void) {
    char b = 38;
    for (int i = 0; i < 100; ++i) {
        b++;
    }

    printf("b = %d\n", b);
}
```

Εφαρμογή Α2 – Μορφοποίηση (1/2)

Το κλειδί είναι πάντα το δυαδικό σύστημα και οι μετατροπές του από/προς το δεκαδικό. Το δεκαεξαδικό σύστημα και το οκταδικό είναι/χρησιμοποιούνται ως ομαδοποιήσεις των ψηφίων του δυαδικού.

Έτσι:

```
          8 4 2 1   8 4 2 1     4 2 1   4 2 1   4 2 1
B4 → 1011 0100 → 010 110 100 → 264 (οκταδικό)
```

Και

```
                unsigned char
               ↙
          +128  32    8 4 2 1
B4 →     1011 0100 → 128+32+16+4 = 180 (δεκαδικό)
            64  16
```

Εφαρμογή Α2 – Μορφοποίηση (2/2)

Εδώ το ζητούμενο είναι διαφορετικό. Έχω μία μεταβλητή τύπου **char** δηλαδή μικρός προσημασμένος ακέραιος 8bit με πεδίο τιμών:

-128 έως +127

Και έχω την αρχική τιμή 38 στην οποία προσθέτω 100 φορές τη μονάδα, δηλαδή το 100.

Θεωρητικά θα ήταν 38 + 100 = 138 όμως αυτό είναι έξω από το πεδίο τιμών. Τι συμβαίνει λοιπόν; Υπερχείλιση!

Η επόμενη τιμή από το +127 είναι το -128, δηλαδή

127+1 = -128

άρα ένας εύκολος τρόπος να σκεφτούμε είναι ότι:

- από το 38 μέχρι το 127 (που είναι το όριο του πεδίου τιμών) είναι 89 μονάδες
- η επόμενη μονάδα κάνει την τιμή μας -128
- και περισσεύουν (από το 100 που προσθέσαμε) και άλλα 10 = 100 - 1 - 89.

Άρα από το -128 μετράμε και άλλα 10 και φτάνουμε στο -118, που είναι και η απάντηση.

Εφαρμογή Α3 – C και γλώσσα μηχανής

Έχουμε τρεις κώδικες C και τους αντίστοιχους κώδικες γλώσσας μηχανής που έχουν προκύψει από το compilation των κωδίκων της C. Ταιριάξτε τους κώδικες βάσει τον αποσπασμάτων που παρουσιάζονται:

```
register short int a;
…
a = c;
…
a++;
…
b = a;
```

A

```
do {
   …
   q = calc(x,y);
   …
   ++q;
   …
} while (q != z);
```

B

```
int a = 100, b = 200;
…
int c = a + b;
…
if (c == d) {
   …
}
```

Γ

```
LOAD R4, 0xE7
…
INC R4
…
JUMP R4, 0x08
```

1

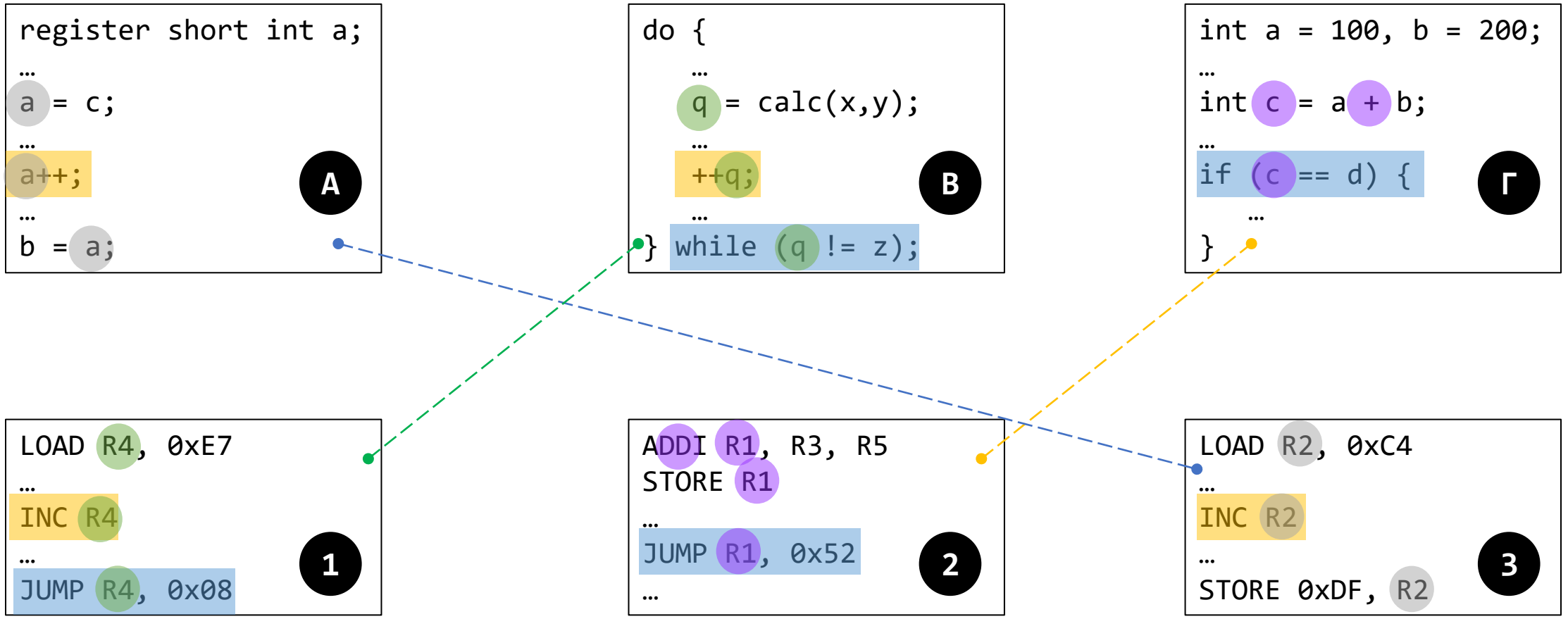
```
ADDI R1, R3, R5
STORE R1
…
JUMP R1, 0x52
…
```

2

```
LOAD R2, 0xC4
…
INC R2
…
STORE 0xDF, R2
```

3

Εφαρμογή Α3 – C και γλώσσα μηχανής



Εφαρμογή Α4 – Γλώσσα μηχανής Ι

Γράψτε τον κατάλληλο κώδικα γλώσσας μηχανής ώστε να μετατρέψετε τον αριθμό που υπάρχει στη θέση μνήμης **F0** στον αντίθετό του, εκφρασμένο σε συμπλήρωμα ως προς 2 (c2) στην ίδια θέση μνήμης.

Εντολή	*Κωδ*	*Τελεστέοι*			*Ενέργεια*
	d_1	d_2	d_3	d_4	
HALT	0				Διακόπτει την εκτέλεση του προγράμματος
LOAD	1	R_D	M_S		$R_D \leftarrow M_S$ (LOAD FROM MEMORY)
STORE	2	M_D		R_S	$M_D \leftarrow R_S$ (STORE TO MEMORY)
ADDI	3	R_D	R_{S1}	R_{S2}	$R_D \leftarrow R_{S1} + R_{S2}$ (ADD INTEGER ή C2)
ADDF	4	R_D	R_{S1}	R_{S2}	$R_D \leftarrow R_{S1} + R_{S2}$ (ADD FLOATING)
MOVE	5	R_D	R_S		$R_D \leftarrow R_S$
NOT	6	R_D	R_S		$R_D \leftarrow R_S$
AND	7	R_D	R_{S1}	R_{S2}	$R_D \leftarrow R_{S1}$ AND R_{S2}
OR	8	R_D	R_{S1}	R_{S2}	$R_D \leftarrow R_{S1}$ OR R_{S2}
XOR	9	R_D	R_{S1}	R_{S2}	$R_D \leftarrow R_{S1}$ XOR R_{S2}
INC	A	R			R ← R + 1
DEC	B	R			R ← R – 1
ROTATE	C	R	n	0 ή 1	Rot_n R *(εκτός ύλης)*
JUMP	D	R	n		Αν $R_0 \neq$ R τότε PC = *n*, διαφορετικά συνέχισε

Εφαρμογή Α4 – Γλώσσα μηχανής Ι

Τα βήματα είναι τα ακόλουθα

Assembly	**Γλώσσα Μηχανής**	**Περιγραφή**
`LOAD R1, F0`	`11F0`	Φέρνω τον αριθμό σε έναν καταχωρητή
`NOT R1, R1`	`6110`	Συμπλήρωμα ως προς 1 = Αντιστρέφω τα bits = (0→1 και 1→0) δηλαδή NOT
`INC R1`	`A100`	Προσθέτω 1
`STORE F0, R1`	`2F01`	Στέλνω πίσω το αποτέλεσμα στη μνήμη
`HALT`	`0000`	Τερματισμός

Ερωτήσεις?

- Διαβάστε τις σημειώσεις, διαβάστε τις διαφάνειες και δείτε τα videos **πριν** ρωτήσετε
- **Συμβουλευτείτε** τη σελίδα ερωταποκρίσεων του μαθήματος

  https://qna.c-programming.allos.gr
- **Στείλτε** τις ερωτήσεις σας πριν και μετά το μάθημα στο

  c-programming-24@allos.gr

- Εάν έχετε **πρόβλημα** με κάποιο κώδικα στείλτε τον κώδικα ως κείμενο με copy/paste. Εάν θεωρείτε ότι επιπλέον βοηθά και ένα στιγμιότυπο οθόνης, είναι καλοδεχούμενο.
- Επαναλαμβάνουμε : Μην στείλετε ποτέ κώδικα ως εικόνα μας είναι παντελώς άχρηστος!