Άσκηση 5 - Βήμα Η

Έστω ότι στη main δηλώνουμε

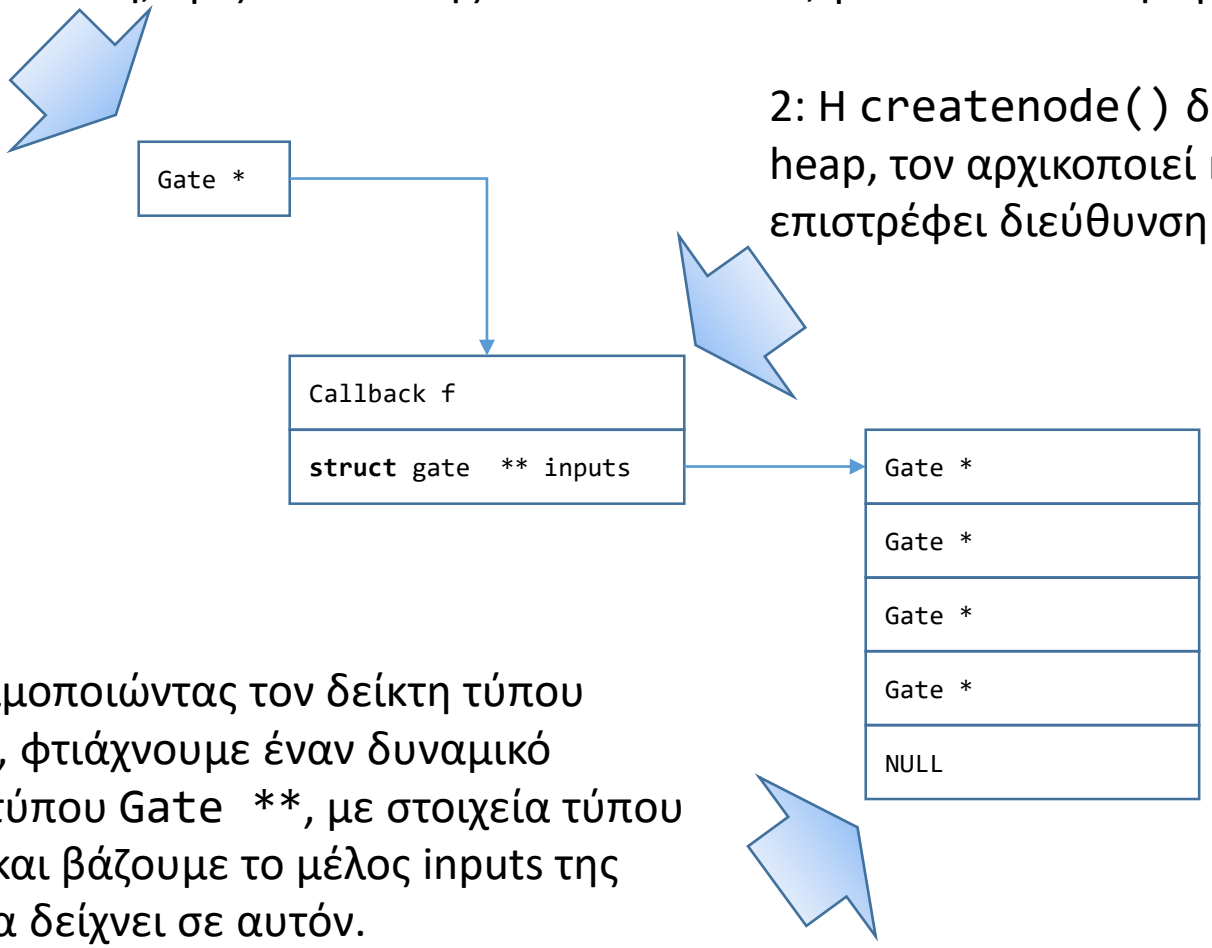
```
Gate * a_ptr, * b_ptr, * c_ptr;

a_ptr = creategate(myand);
b_ptr = creategate(myand);
c_ptr = creategate(myor);
```

- a_ptr, b_ptr, c_ptr είναι δείκτες σε Gate
- Είναι μεταβλητές και αποθηκεύονται στο stack
- Είναι μεταβλητές και λαμβάνουν ως τιμές διευθύνσεις περιοχών που βρίσκονται στο heap
- Η κατάλληλη δέσμευση περιοχής για την αποθήκευση κάθε Gate γίνεται από την creategate()
- Κάθε Gate πρέπει να δημιουργείται μόνο μια φορά
- Χρησιμοποιούμε τον αντίστοιχο δείκτη (τύπου Gate *) για να αναφερθούμε σε κάθε πύλη.

Για κάθε πύλη

1: Για κάθε πύλη, ορίζεται δείκτης τύπου `Gate *`, γίνεται ανάθεση τιμής στη main.



2: Η `createnode()` δεσμεύει χώρο στο heap, τον αρχικοποιεί κατάλληλα, και επιστρέφει διεύθυνση σε αυτόν.

3: Χρησιμοποιώντας τον δείκτη τύπου `Gate *`, φτιάχνουμε έναν δυναμικό πίνακα τύπου `Gate **`, με στοιχεία τύπου Gate *, και βάζουμε το μέλος inputs της πύλης να δείχνει σε αυτόν.
Τερματίζουμε με NULL, υπάρχουν και άλλοι τρόποι.

Σύνδεση πολλών εισόδων

Χρησιμοποιούμε το inputs για να αναφερθούμε στο σύνολο των εισόδων.
Κάθε μέλος έχει ως τιμή τη διεύθυνση **υπάρχουσας** πύλης.
Παράδειγμα

```
(a_ptr -> inputs) [0] = b_ptr;
(a_ptr -> inputs) [1] = c_ptr;
(a_ptr -> inputs) [2] = d_ptr;
(a_ptr -> inputs) [3] = NULL;
```

Θα πρέπει να αρχικοποιήσουμε το inputs με διεύθυνση δεσμευμένου χώρου στο heap που να χωράει τέσσερα στοιχεία τύπου Gate * (για το παραπάνω παράδειγμα)

Αντιπαραβάλλετε το παράδειγμα αυτό, με το στυλ που χρησιμοποιήσαμε στο βήμα Z

```
a_ptr->in1 = b_ptr;
a_ptr->in2 = c_ptr;
```

Φυσικά υπάρχουν πολλοί άλλοι τρόποι!