Bases de programmation – Cours 6. Jeu d'adresses.

Pierre Boudes

8 décembre 2014

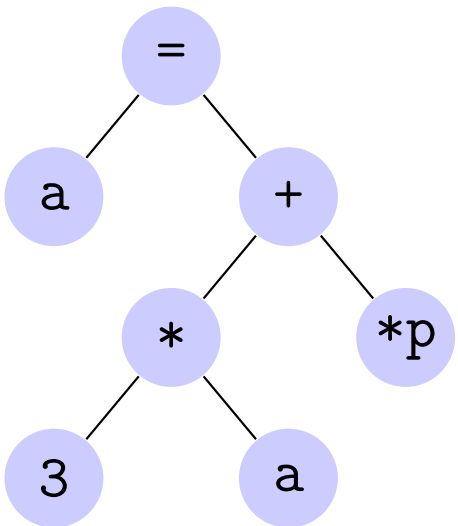
Adresses mémoires, pointeurs

- Si `x` est une variable, `&x` est son adresse mémoire.
- Si `p` contient une adresse mémoire, `*p` est la valeur contenue à cette adresse. Et on dit que `p` est un pointeur car son rôle est de désigner (pointer en direction de) une autre case mémoire.

```
1  int main () {
2    int a = 3;
3    int b = 4;
4    int *p; /* /!\ declaration d'un pointeur */
5    p = &a; /* p contient l'adresse de a */
6    *p = 1; /* ecrit a l'adresse contenue dans p */
7    p = &b;
8    *p = 2;
9    printf("%d␣%d\n", a, b);
10   return EXIT_SUCCESS;
11 }
```

Indirection

```
int a = 5;
int b = 4;
int *p = &b;
a = 3 * a + *p;
```




```
1   valeur 3 r0
2   lecture 10 r1
3   mult r0 r1
4   lecture 12 r2
5   lecture *r2 r3
6   add r1 r3
7   ecriture r3 10
8   stop
9
10  5   # a
11  4   # b
12  11  # p
```

Erreurs de casting

erreurs makes pointer form integer without a cast.

Chaînes

Pour les chaînes `char * s` plutôt que `char s[]`.

```
1  #include <stdio.h>
2
3  typedef char mstring[];
4  typedef char *istring;
5  int main () {
6    mstring toto = "Salut␣mon␣Toto";
7    istring tata = "Bonjour␣Tata";
8    char * personne;
9    personne = tata;
10   printf("%s\n", personne);
11   // personne[5] = '\0'; <-- bus error 10
12   personne = toto;
13
14   printf("%s\n", personne);
15   personne[5] = '\0';
16   printf("%s\n", personne);
17   printf("%s\n", toto);
```

Argument résultat

```
1  int age = 0;
2  scanf("%d", &age);
```

Focus sur le passage de valeurs

Les fonctions communiquent **des valeurs**, pas des noms de variables.

```
1  void permute_valeurs(int a,int b);
2
3  int main() {
4      int x = 1;
5      int y = 2;
6      permute_valeurs(x, y);
7      printf("x␣=␣%d␣et␣y␣=␣%d\n",x,y);
8      return EXIT_SUCCESS;
9  }
10
11 void permute_valeurs(int a, int b) {
12     int aux;
13     aux = a;
14     a = b;
15     b = aux;
16 }
```

Focus sur le passage de valeurs

Les fonctions communiquent **des valeurs**, pas des noms de variables.

```
1  void permute_valeurs(int a,int b);
2
3  int main() {
4       int x = 1;
5       int y = 2;
6       permute_valeurs(&x, &y);
7       printf("x␣=␣%d␣et␣y␣=␣%d\n",x,y);
8       return EXIT_SUCCESS;
9  }
10
11 void permute_valeurs(int *a, int *b) {
12      int aux;
13      aux = *a;
14      *a = *b;
15      *b = aux;
16 }
```

Passage de tableaux en paramètre

Lorsqu'on passe un tableau en paramètre d'une fonction le paramètre formel qui le reçoit doit être un pointeur (du bon type).

Passage de structure par adresse

Pour rendre les appels plus rapides on passe souvent juste l'adresse d'une structure en paramètre d'une fonction plutôt que de passer toute la structure. Cela évite la copie de la structure.
Nouvelle notation (pratique) `->` : si `fiche` est un pointeur sur une structure, plutôt que d'écrire `(*fiche).nom` on écrira `fiche->nom`.

Allocation mémoire

malloc / free