Functions & Linked List – Devoir N°2

NB. les schémas doivent être dessinés à l'aide d'un logiciel

Exercice 1 :

Soit la structure de données suivante :

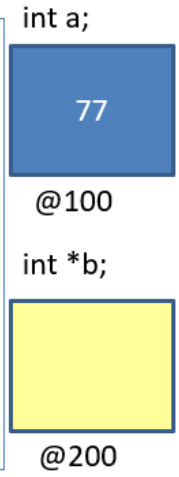
```
typedef struct element{
int data;
struct element *next;
}Element;
```

Dans le reste de l'énoncé, nous allons utiliser cette même structure de données.

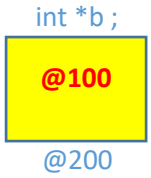
Exercice 1 :

#program1

```
11  int main()
12  {
13  int a;
14  int *b;
15  a=77;
16  b=&a;
17  f(b);
18  printf("%p %p %p \n",&b,b,&a);
19  printf("%d %d",a,*b);
20  return 0;
21  }
```



1) Suite au code source ci-dessus, renseigner le contenu de la case jaune

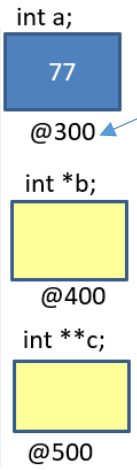


2) Donner le résultat affiché par le programme

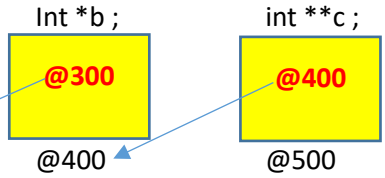
Affichage : @200 @100 @100
77 77

#program2

```
11  int main()
12  {
13  int a;
14  int *b;
15  int **c;
16  a=77;
17  b=&a;
18  c=&b;
19  printf("%p %p %p %p \n",&c,c,*c,b);
20  printf("%d %d %d",**c,*b,a);
21  return 0;
22  }
```



3) Suite au code source ci-dessus, renseigner le contenu des cases jaunes



4) Donner le résultat affiché par le programme

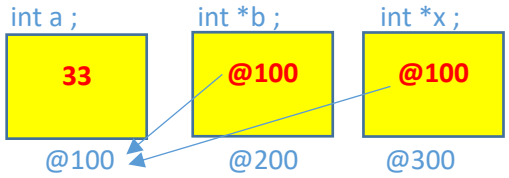
Affichage : @500 @400 @300 @300
77 77 77

Exercice 2

#program3

```
 8  void g1(int *x){
 9  *x=33;
10  }
11  void g2(int *x){
12  x=(int*)malloc(sizeof(int));
13  *x=33;
14  }
15  int main()
16  {
17  int a;a=77;
18  int *b;
19  b=&a;
20  printf("%d %d \n --- \n",a,*b);
21  g1(b);
22  printf("%d %d",a,*b);
23  return 0;
24  }
```

1) Donner le schéma de mémoire des variables qui correspondent aux deux fonctions main() et g1()

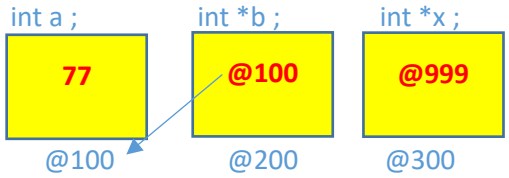


2) Donner le résultat du programme

Affichage :

```
77 77
 ---
33 33
```

3) Modifier g1(b) par g2(b),
Donner le schéma de mémoire des variables qui correspondent aux deux fonctions main() et g2()



4) Donner le résultat du programme

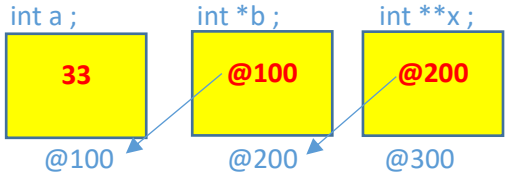
Affichage :

```
77 77
 ---
77 77
```

#program4

```
 3  void h1(int **x){
 4  **x=33;}
 5  void h2(int **x){
 6  *x=(int*)malloc(sizeof(int));
 7  **x=33;}
 8  void h3(int **x){
 9  x=(int**)malloc(sizeof(int*));
10  *x=(int*)malloc(sizeof(int));
11  **x=33;}
12  int main()
13  {
14  int a;a=77;
15  int *b; b=&a;
16  printf("%d %d \n --- \n",a,*b);
17  h1(&b);
18  printf("%d %d",a,*b);
19  return 0;
20  }
```

1) Donner le schéma de mémoire des variables qui correspondent aux deux fonctions main() et h1(&b)

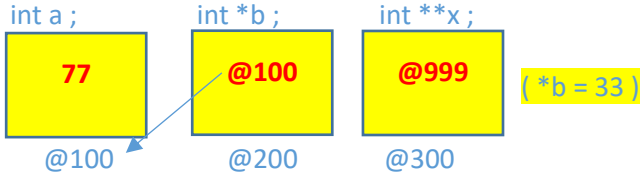


2) Donner le résultat du programme

Affichage :

```
77 77
 ---
33 33
```

3) Modifier h1(&b) par h2(&b),
Donner le schéma de mémoire des variables qui correspondent aux deux fonctions main() et h2(&b)

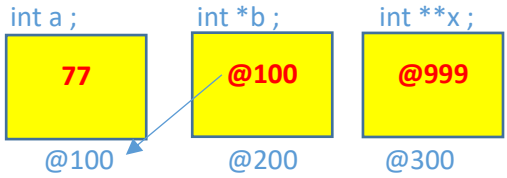


4)Donner le résultat du programme

Affichage :

```
77 77
---
77 33
```

5) Modifier h1(&b) par h3(&b),
Donner le schéma de mémoire des variables qui correspondent aux deux fonctions main() et h3(&b)



6) Donner le résultat du programme

Affichage :

```
77 77
---
77 77
```

Dans la suite, nous allons utiliser les éléments suivants :

```
typedef struct element{
int data;
struct element *next;
}Element;
```

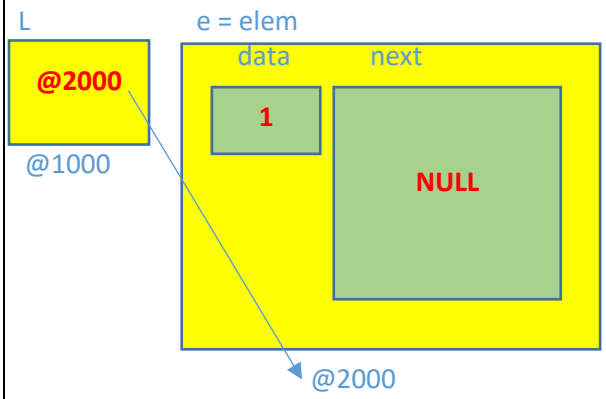
```
Element *createElement(int data){
Element *elem;
elem=(Element*)malloc(sizeof(Element));
elem->data=data;
elem->next=NULL;
return elem;
}
```

```
void displayList(Element *L){
   while(L!=NULL){
      printf("%d \n",L->data);
      L=L->next;
   }
}
```

Exercice 3

```
#program5
27  void addVersion1(int data, Element *L){
28  Element *e;
29  e=createElement(data);
30  if(L==NULL){
31  L=e;
32  }
33  }
34  int main()
35  {
36  Element *L;
37  L=NULL;
38  addVersion1(1,&L);
39  displayList(L);
40  return 0;
41  }
```

1) Donner le schéma de mémoire des variables qui correspondent aux deux fonctions main() et addVersion1(1,&L)



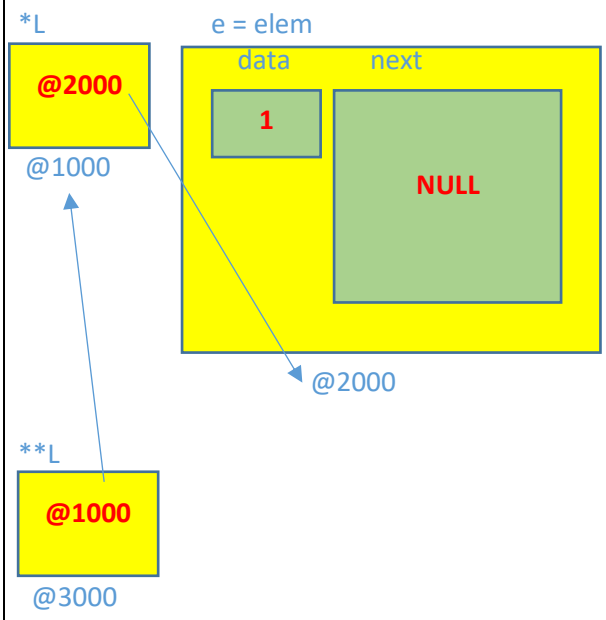
2) Donner le résultat du programme

Affichage : 1

#program6

```
27  void addVersion2(int data, Element **L){
28  Element *e;
29  e=createElement(data);
30  if(*L==NULL){
31  *L=e;
32  }
33  }
34  int main()
35  {
36  Element *L;
37  L=NULL;
38  addVersion2(1,&L);
39  displayList(L);
40  return 0;
41  }
```

1) Donner le schéma de mémoire des variables qui correspondent aux deux fonctions main() et addVersion2(1,&L)



2) Donner le résultat du programme

Affichage : 1

3) si on ajoute la ligne addVersion2(2,&L), est ce qu'un nouveau éléments sera inséré dans la liste

Non, parce que L non plus égal a NULL. Donc le block if {} ne va pas être exécuté