Langage procédural

Le langage C

Nizar OUARTI

Laboratoire ISIR (email: ouarti@isir.upmc.fr)

2010 2011

Les blocs

- Un bloc est une suite d'instructions
- Ils est délimité par des accolades
- Usage des accolades { **instruction1 ; instruction2 ; instruction3** }

Les conditions logiques

- Expressions logiques simples
  - != différent de
  - == égalité logique
  - >= supérieur ou égal
  - > supérieur
  - ! non logique
  - ex : (a<b)
- Expressions logiques composées
  - && ET logique
  - || OU logique
  - $(n < b)$ && $(n > c)$ n inférieur à b et n supérieur à c

Les opérateurs conditionnels

- Servent à contrôler le programme
- Ils permettent d'avoir un programme qui s'exécute différemment en fonction des données
- mots clé {**if else**} {**switch case**}

L'instruction if else (si sinon)

- L'opérateur sert à déterminer si une condition est vrai
- Si c'est le cas les instructions de son bloc sont exécutées
- Sinon ce sont les instructions du bloc du else qui le sont

```
1  /* Définiton formelle */
2  if (CONDITION) //teste si la condition est vraie
3  {
4          INSTRUCTION1;// Les instruction suivantes
5          INSTRUCTION2;// s'exécutent si la condition
6          INSTRUCTION3;// est vrai
7  }
8  else
9  {
10         INSTRUCTION1;// Les instruction suivantes
11         INSTRUCTION2;// s'exécutent si la condition
12         INSTRUCTION3;// n'est pas vrai
13 }
```

L'instruction if else (si sinon)

```
1  /* Exemple concret */
2  if (N>5) //Si la condition est vraie
3  {
4          N=N+1;// les instruction suivantes s'exécutent
5          printf("la valeur de N est : %d",N);
6  }
7  else     //Sinon
8  {
9          N=0;// Les instruction suivantes s'éxécutent
10         printf("la valeur de N est : %d",N);
11 }
```

Ambiguité avec else

- else va toujours avec le if le plus proche s'il n'y a pas d'accolades
- Ne pas mettre d'espace entrre le else et le bloc du if

```
1 //Ecriture ambigue à éviter
2 if (n>0) //Si la condition est vraie
3         if(x>y)
4                 z=y;
5         else    //Ce else dépend du if juste au dessus
6                 z=x;
```

Ecrire clairement

- Il vaut mieux toujours mettre des accolades
- Cela rend le programme moins ambigu et permet d'éviter des erreurs
- Ne pas oublier d'indenter
- En quoi le deuxième programme diffère du premier ?

```
1 //Ecriture non ambigue avec accolades
2 if (n>0) //Si la condition est vraie
3 {
4         if(x>y)
5                 z=y;
6 }
7 else     //Ce else dépend du if juste au dessus
8         z=x;
```

L'instruction **else if**

- Cela sert dans le cas où on a plusieurs conditions a tester
- Toutes les conditions sont testés
- Si aucune ne marchent le bloc du dernier else sera exécuté

L'instruction **else if**

```c
/* Exemple concret */
if (N==1) //Si la condition est vraie
{
        printf("la valeur de N est : %d",N);
}
else if (N==2) //Si la condition est vraie
{
        printf("la valeur de N est : %d",N);
}
else if (N==3) //Si la condition est vraie
{
        printf("la valeur de N est : %d",N);
}
else    //Sinon si N n'est aucune des autre valeurs
{
        printf("la valeur de N %d n'est pas autorisée",
            N);
}
```

L'instruction **switch case**

- C'est la méthode conseillé dans le cas où on a plusieurs conditions a tester
- Toutes les conditions sont testés
- Si aucune ne marchent, le bloc default sera exécuté
- La variable peut être : char, short, int, long (signé ou non)
- La variable ne peut être un nombre à virgule

L'instruction **switch case**

```
1  /* Définiton formelle */
2  switch (variable) // Variable dont on teste les valeurs
3  {
4          case VALUE1 :
5          {
6                  INSTRUCTION1;// s'exécute si variable est égal à VALUE1
7          }
8          case VALUE2 :
9          {
10                 INSTRUCTION2;// s'exécute si variable est égal à VALUE2
11         }
12         case VALUE2 :
13         {
14                 INSTRUCTION3;// s'exécute si variable est égal à VALUE3
15         }
16         default :INSTRUCTION4;// s'exécute si variable n'est égal à aucunes des
               VALUEi
17 }
```

L'instruction switch case

```
1  /* Exemple concret */
2  switch (N) //Teste plusieurs conditions
3  {
4          case 1:
5          {
6                  printf("La taille est petite");
7                  break; //on sort du switch
8          }
9          case 2:
10         {
11                 printf("La taille est moyenne");
12                 break; //on sort du switch
13         }
14         case 3:
15         {
16                 printf("La taille est grande");
17                 break; //on sort du switch
18         }
19         default:
20         {
21                 printf("Cette valeur n'est pas autorisée"N);
22                 break; //on sort du switch
23         }
24 }
```

Les Boucles

- Comme leur nom l'indique elles servent à répéter plusieurs fois une partie du programme
- Elles peuvent être déterminées ou indéterminées
- Dans un cas on sait combien de fois sera répété le bloc (déterminé)
- Dans un cas on ne sait pas combien de fois sera répété le bloc (indéterminé)
- mots clé {**for**} {**while**} {**do while**}

Boucles déterminées **for**

- L'instruction sert exécuter plusieurs fois le même code
- Cette exécution est faite un nombre déterminé de fois

```
1 /* Définiton formelle */
2 for (INITIALISATION;CONDITION;INCREMENTATION)
3 {
4         INSTRUCTION1;// Les instruction suivantes
5         INSTRUCTION2;// s'exécutent N fois
6         INSTRUCTION3;//
7 }
```

L'instruction for

```
1 /* Exemple concret */
2 for (i=0;i<N;i++)
3 {
4         tmp=tmp+1; // Les instruction suivantes
5         printf("La valeur de tmp est: %d",tmp); // s'
              exécutent N fois
6 }
```

Boucles indéterminées **while**

- L'instruction sert exécuter plusieurs fois le même code
- Cette exécution est faite un nombre indéterminé de fois

```
1 /* Définiton formelle */
2 while (CONDITION)
3 {
4         INSTRUCTION1;// Les instruction suivantes
5         INSTRUCTION2;// s'exécutent tant que la
6         INSTRUCTION3;// condition est vrai
7         modification_conditions();
8 }
```

L'instruction while

```
1 /* Exemple concret */
2 i=0;// initialisation
3 Temps_max=25;
4 while(i<Temp_max)
5 {
6         i=mesure_temperature(); // Les instructions
              suivantes
7         printf("La température n'est pas trop élevée");
              // s'exécutent N fois
8 }
```

différence entre for et while

- Il est possible d'interchanger while et for, toutefois...
- Il est conseillé d'utiliser des boucles for lorsque l'on sait très bien quel nombre d'itérations doivent être faite
- Il est conseillé d'utiliser des boucles while quand on ne connait pas à l'avance la fin des itérations mais plutôt une condition de fin

Boucles indéterminées **do while**

- L'instruction sert exécuter plusieurs fois le même code
- Cette exécution est faite un nombre indéterminé de fois
- la différence avec l'instruction while toute seule est que cette fois le test se fait après l'exécution du bloc

```
1 /* Exemple concret */
2 i=0;// initialisation
3 Temps_max=25;
4 do
5 {
6         i=mesure_temperature(); // Les instruction
              suivantes
7         printf("La température n'est pas trop élevée");
              // s'exécutent N fois
8 }
9 while(i<Temp_max);
```

L'instruction break

- Sert à sortir des boucles
- Pour sortir des boucles for, while ou do while
- Sert à ajouter une condition de terminaison supplémentaire

Problème

- Vous allez concevoir un pilote automatique
- Le programme doit en permanence tester ces différent aspects
- Toutes ces choses seront faites un peu comme une "tâche de fond"
- Celui doit vérifier plusieurs choses
- Si un bouton est enfoncé si oui lequel
- Si l'altitude n'est pas en dessous d'un seuil
- Si aucun angle de l'attitude est supérieur à un trop gros angle
- Si le niveau du carburant est supérieur au nombres de litres requis pour faire les kilomètres restant
- Il doit vérifier l'état de fonctionnement des 4 moteurs

avion

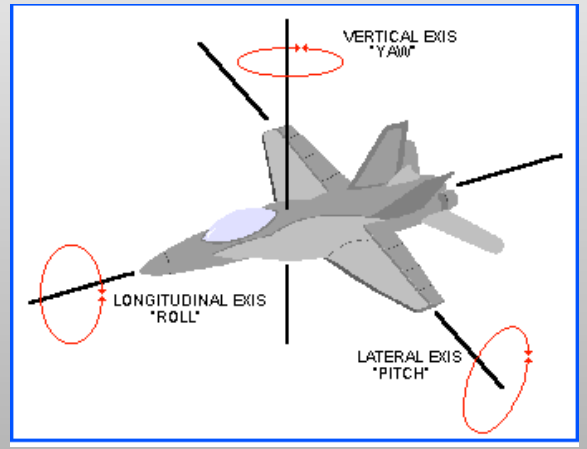

FIGURE: Différents angles d'attitude

- Des information venant d'un gyromètre sont disponibles
- Donc des vitesses angulaires
- Il ne faut pas dépasser 30° d'angle
- Nous avons aussi un altimètre qui donne l'altitude en mètre
- Il ne faut pas être plus bas que (500 mètres d'altitude
- Ces deux évènements sont associés à une alerte spécifique

Le carburant

- Il suffit de tester la jauge
- Sachant que l'avion consomme K1 litre par kilomètres
- Sachant que la distance à parcourir est de K2 kilomètres
- Calculer s'il reste sufisament de carburant ou sinon donner une alerte

Les 4 moteurs

- Il faut tester les paramètres des moteurs les uns après les autres
- paramètres : température, problème d'électronique, problème mécanique

Les boutons

- Il ne peuvent être allumés qu'un à la fois
- ils ont différentes couleurs
  - bleu
  - rouge
  - vert
  - jaune
- Chacun étant lié à une action
  - Arrêt du pilote automatique
  - Ouverture des vannes des réserves d'essences
  - Allumer les lumières extérieures
  - Ouverture de la porte du cockpit