Anexo A. Estilo de tabulación

Introducción

En este anexo introducimos una serie de reglas básicas para adquirir un buen estilo de tabulación del programa. Un programa, además de ser efectivo, rápido y estar bien estructurado, debe estar bien tabulado. De esta forma, hacemos que el programa sea más legible por otra persona y por nosotros mismos, facilitando su lectura, depurado y reutilización.

Las reglas de tabulación que vamos a relatar no son las únicas que existen. Además, aunque se deben aplicar de la forma más estricta posible, podríamos relajarlas por ciertas razones concretas de espacio o estilo.

Comenzaremos el anexo relatando las normas generales de tabulación, para después puntualizar ciertos casos excepcionales muy generalizados.

Reglas básicas de tabulación

En esta sección relataremos un conjunto de reglas básicas de tabulación. Comenzaremos por la regla para tabular bloques, que como veremos se extiende a todas las estructuras.

Bloques y sub-bloques

Se trata de la regla más importante. Cuando creamos un nuevo bloque, el contenido del mismo debe contener una marca de tabulación más que el bloque en el que está incluido. De igual forma, al cerrar un bloque tenemos que bajar una marca de tabulación. La única razón por la que se aumenta/disminuye la profundidad de tabulado es la apertura/cierre respectivamente de bloques. Ejemplo:

```
//librerias y constantes
#include <stdio.h>
#include <stdlib.h>

//Hasta ahora estamos en el bloque global, luego
//todo va sin tabular
int main(void)
```

```
{
    //main define un nuevo bloque, por lo que todo su cuerpo
    //se tabula a la derecha
    int a;
    for(a=1;a<10;a++){
        //la estructura for define un nuevo bloque, luego
        //todas las instrucciones del mismo se tabulan
        //una unidad más
        printf("Hola\n");
        if(a<5){
            //lo mismo sucede con el bloque del condicional
            printf("mensaje");
        }
        //al cerrar el bloque del condicional disminuimos una
        //tabulación
    }
    //lo mismo sucede con el fin del bloque del for
    return(0);
}
```

En general, la llave de apertura del bloque se escribe a la derecha de la cabecera de la estructura y la llave de cierre debajo de la primera letra de dicha cabecera, coincidiendo con la marca de tabulación anterior:

```
//bloque actual
if(a<0){ //llave de apertura a la derecha de la cabecera
    //instrucciones del nuevo bloque tabuladas una unidad más
    printf("hola\n");
}//llave de cierre en la marca de tabulación anterior
//resto de instrucciones del bloque actual
```

Estructuras de decisión

Teniendo en cuenta la regla anterior, el condicional simple se escribe:

```
if([condicion]){
    //bloque de instrucciones del SI tabulado una unidad más
}
```

En caso de tener parte else, hemos de tener en cuenta que se cierra el bloque del SI y se abre el del SINO, por tanto, ambos bloques están a la misma profundidad de tabulación. Las llaves de apertura y cierre así como la instrucción else, se escriben en la misma línea:

```
if([condicion]){
    //bloque del SI, tabulado una unidad más
}else{
    //bloque del SINO, tabulado una unidad más
}
```

En el siguiente ejemplo podemos ver cómo se escribirían dos condicionales simples anidados:

```
if(a>0){
    printf("Positivo\n");
}else{
    printf("Negativo o cero\n");
    if(a==0){
        printf("Definitivamente cero\n");
    }else{
        printf("Definitivamente negativo\n");
    }
}
```

En el caso del condicional múltiple (switch), las instrucciones asociadas a los casos y el break de cada caso deben contener una tabulación, así que se tabulan como si fuesen bloques. Los case quedan a la altura de switch porque en realidad son etiquetas, no instrucciones. Ejemplo:

```
int i;
printf("Introduzca numero:");
scanf("%d",&i);
switch(i){
case 0:
    //Instrucciones del caso 0
    printf("caso 0\n");
    break;
case 1:
    //Instrucciones del caso 1
    brintf("caso 1\n");
    break;
case 2:
    //Instrucciones del caso 2
    printf("caso 2\n");
    break;
default:
    //Instrucciones del default
    printf("Caso por defecto\n");
    break;
}
```

Estructuras de repetición

Las estructuras de repetición son especialmente sencillas de escribir a partir de la regla de bloques y teniendo en cuenta que la llave de apertura se coloca en la misma línea y a la derecha de la cabecera de la estructura. Por otro lado, la llave de cierre se coloca

debajo de la primera letra de la cabecera, o lo que es lo mismo, en la marca de tabulación del bloque en el que se encuentra.

La estructura de repetición con condición inicial se escribe:

```
while([condicion]){
    //instrucciones del bloque con una
    //profundidad más de tabulación
}
```

Por otro lado, la estructura de repetición con condición final se escribe:

```
do{
    //instrucciones del bloque con una
    //profundidad más de tabulación
}while([condición]);
```

Finalmente, la estructura de repetición con contador se escribe:

```
for([inicio];[condicion];[incremento]){
    //instrucciones del bloque con una
    //profundidad más de tabulación
}
```

A continuación, presentamos un ejemplo:

```
int a,b,c;
for(a=0;a<10;a++){
    printf("Introduzca un numero:");
    scanf("%d",&a);
    while(a>0){
        printf("%d\n",a);
        a=a-1;
    }
    do{
        printf("Introduzca numero negativo:");
        scanf("%d",&b);
    }while(b>0);
    printf("Su positivo es %d\n",-b);
}
//resto del código…
```

Funciones

A la hora de escribir una función, el tipo de salida se coloca sin tabular y en la siguiente línea el resto de la cabecera (también sin tabular) seguida de la llave de apertura del bloque. El bloque se tabula como todos los casos anteriores (incluyendo la instrucción return, que se tabula como el resto). Ejemplo:

```
int mayor(int a, int b){
    int r;
    if(a>b){
```

```
                r=a;
        }else{
                r=b;
        }
        return(r);
}
int  main(void){
        int a,b,c;
        a=5;
        b=7;
        c=mayor(a,b);
        return(0);
}
```

Consideraciones adicionales

- La única forma de tabular es utilizando el tabulador (TAB). Nunca se debe utilizar la barra espaciadora para cuadrar el programa.

- Como norma general, los comentarios se tabulan igual que si fueran instrucciones del programa, respetando la profundidad de tabulación en la que se encuentran. Lo mismo se puede decir de la creación de variables. Ejemplo:

```
//librerias y constantes
#include <stdio.h>
#include <stdlib.h>

#define PI 3.14
#define K 7

// Comentarios relacionados con la funcion
int Funcion(void){
        //comentario relacionado con las variables
        int v1,v2;
        //comentario relacionado con el for
        for(v1=5;v1<10;v1++){
                //comentario relacionado con el printf
                printf("Mensaje\n");
        }
        return(0);
}
```

- Todo lo que no esté dentro de una función no se tabula (se deja pegado al borde izquierdo de la página): constantes, inclusión de librerías, comentarios globales, etc. (ver ejemplo anterior).

Algunas excepciones

A continuación, presentamos algunas excepciones, a modo de ejemplo, para tabular ciertas estructuras.

Bloques de una sola instrucción

Un bloque se define como una sola instrucción o varias enmarcadas entre llaves. Por tanto, cuando el bloque de una estructura solo tiene una instrucción se suele poner sin llaves, y colocada a la derecha de la cabecera de la estructura. Por ejemplo, un condicional simple como este:

```
if(a>b){
    printf("A es mayor que B\n");
}else{
    printf("B es mayor o igual que A\n");
}
```

se podría escribir:

```
if(a>b) printf("A es mayor que B\n");
else printf("B es mayor o igual que A\n");
```

Como vemos, resulta una escritura más compacta. El mismo procedimiento se puede realizar con bucles, como en el siguiente ejemplo:

```
for(i=0;i<10;i++){
    printf("%d\n",i);
}
```

que quedaría en una sola línea:

```
for(i=0;i<10;i++) printf("%d\n",i);
```

Estructuras de if-else anidados

En determinados programas se recurre a una estructura compuesta por varios if-else anidados, por ejemplo:

```
if(a<5){
    printf("Menor a 5\n");
}else{
    if(a<10){
        printf("Menor a 10\n");
    }else{
        if(a<15){
            printf("Menor a 15\n");
        }else{
            if(a<20){
                printf("Menor a 20\n");
            }else{
```

```
                                        printf("Mayor o igual a 20\n");
                                    }
                                }
                            }
                        }
```

Como vemos, aunque hemos seguido las reglas de tabulación, la profundidad de los if-else anidados es muy alta, lo cual puede llegar a ser incómodo para leer el programa. Normalmente, en este tipo de estructuras se obvian las llaves del bloque del else y la cabecera de cada if se escribe en la misma línea del else correspondiente. De esta forma, la estructura quedaría:

```
if(a<5){
    printf("Menor a 5\n");
}else if(a<10){
    printf("Menor a 10\n");
}else if(a<15){
    printf("Menor a 15\n");
}else if(a<20){
    printf("Menor a 20\n");
}else{
    printf("Mayor o igual a 20\n");
}
```

Como podemos ver, la estructura queda mucho más clara. Al eliminar las llaves de los else, reducimos los niveles de profundidad, lo cual deja ver mejor la estructura.