Funciones de Ingreso y Salida Estándar – Detener la Ejecución

Funciones cin, cout, getch(), getche()

Función cin cout

cin, toma caracteres de la entrada estándar (teclado)

cout, muestra caracteres en la salida estándar (pantalla)

Para utilizar las función cin/cout es necesario incluir la directiva: **#include <iostream>**

Estas funciones se utilizan mediante los operadores **<<** y **>>.** El operador << se denomina operador de inserción; y apunta al objeto donde tiene que enviar la información. Por lo tanto la sintaxis de cout será:

cout<<variable1<<variable2<<...<<variablen;

No olvidar que las cadenas de texto son variables y se ponen entre " " (comillas dobles).

Por su parte >> se denomina operador de extracción, lee información del flujo cin (a la izquierda del operador) y las almacena en las variables indicadas a la derecha).

La sintaxis sería la siguiente:

cin>>variable1>>...>>variablen;

Funciones getch() y getche()

Si se necesita que el usuario introduzca un carácter por el teclado se usan las funciones *getch* *getche*. Estas esperan a que el usuario introduzca un carácter por el teclado. La diferencia entre *getche* y *getch* es que la primera saca por pantalla la tecla que hemos pulsado y la segunda no (la e del final se refiere a *echo*=eco).

Para utilizar la función printf es necesario incluir la directiva: **#include <conio>**

Sintaxis

getch();
getche();

Función printf

Función printf

Toma como argumento una cadena de caracteres y lo imprime en pantalla, este argumento debe encerrarse entre comillas dobles " ".

Para utilizar la función printf es necesario incluir la directiva: **#include <stdio>**

Sintaxis

printf("prueba");

Símbolos que se suelen utilizar en el printf

\a = Alerta

\b = Espacio atrás

\f = Salto de página

\n = Salto de línea

\r = Retorno de carro

\t = Tabulación horizontal

\v = Tabulación vertical

Importante: los comentarios se agregan entre /* y */, por lo que no son considerados al momento de la compilación. Para comentar una sola línea se puede utilizar //.

Sintaxis

printf ("utilizando el salto de línea \n"); // la próxima cadena de carácter la imprime en la línea // de abajo

printf ("utilizando la alarma \a"); /*emite un sonido después de imprimir la cadena de carácter*/

Además de la cadena de carácter que se puede imprimir se puede utilizar para visualizar el contenido de una variable, para esto se debe realizar la especificación de conversión. Cada especificación de conversión comienza con un % y termina con un carácter de conversión. Si el carácter después del % no es una especificación de conversión, el comportamiento no esta definido.

Función printf

Sintaxis

```
…..
  int var_nro;  /* se define una variable de tipo entero */
  printf("el nro es:   %i \n" var_nro);  /* el mensaje y el contenido de la variable var_nro.*/
…….
```

Conversiones básicas del printf

CODIGO	FORMATO
%c	Un solo carácter
%d	Decimal (un entero)
%i	Un entero
%f	Punto decimal flotante
%e	Notación científica
%o	Octal
%x	Hexadecimal
%u	Entero sin signo
%s	Cadena de caracteres
%%	Imprime un signo %
%p	Dirección de un puntero

Ejemplos de especificaciones de conversión, se utiliza

- %6d para enteros de 6 caracteres de amplitud
- %f para decimales
- %6df para decimales de 6 caracteres de amplitud
- %.2f para decimales con 2 caracteres después del punto decimal
- %6.2f para decimales con 6 enteros y 2 decimales.

Función scanf

Función scanf

Es muy similar al de printf con una diferencia, da la posibilidad de que el usuario introduzca datos desde el teclado en vez de mostrarlos. No permite mostrar texto en la pantalla, por lo que debe acompañarse con un printf delante.

El número de parámetros puede variar, pero el primero es una cadena que especifica los formatos de los datos a ingresar.

Sintaxis

```
int a;
scanf("%i", &a); /* esto permite ingresar un valor para la variable a del tipo entero */
```

Podemos preguntar por más de una variable a la vez en un sólo scanf, hay que poner un %i por cada variable

```
int a,b,c;
scanf( "%i %i %i", &a, &b, &c );
printf( "Han tecleado los números %i %i %i\n", a, b, c );
```

Todas las variables usadas para recibir valores a través de scanf(), deben ser pasadas por su direcciones, lo cual significa que todos los argumentos deben apuntar a las variables que los van a contener.

La función scanf() necesita conocer la dirección de memoria de las variables para poder "cargar" el dato ingresado, es eso por que se coloca el símbolo **&** delante de las variables, ya que éste es un operador unario que precediendo al nombre de las variables indica que nos referimos a la dirección de la misma y no a su contenido.