http://www.nahuelsanchez.com/16/problemas-al-usar-el-gets-y-scanf/

Dos funciones muy utilizadas en C o C++ son `scanf` (para almacenar enteros, flotantes y caracteres) y `gets` (para almacenar una cadena de caracteres en un arreglo de caracteres previamente definido), las cuales provocan un curioso "error" cuando se utilizan seguidas en un mismo programa.

El siguiente código en C corresponde a un pequeño programa de ejemplo en el cual primero se pide ingresar un entero positivo para almacenarlo en la variable entera "entero" mediante la función `scanf`, y luego se pide ingresar una cadena de caracteres para ser almacenada en el arreglo de caracteres "arreglo[10]" utilizando la función `gets`.

```
main(){
int entero;
char arreglo[10];

printf("Ingrese un entero positivo: ");
scanf("%d",&amp;entero);
printf("\nIngrese una cadena de caracteres: ");
gets(arreglo);

printf("\n\nUsted ingreso %d como Entero y %s como cadena de
caracteres\n",entero,arreglo);
system("pause");
}
```

Aunque el código no contenga errores, si lo ejecutamos nos daremos cuenta que cuando el programa llega a la línea del `gets`, en vez de ejecutarla se la saltea, deja un espacio en blanco y pasa a la siguiente (en este caso muestra el `printf` con los valores almacenados en las variables).

Este comportamiento se debe a que cuando el programa nos pide ingresar el entero positivo y nosotros lo tipeamos, luego al presionar Enter enviamos no sólo el número escrito sino también el código correspondiente a ese salto de línea (el Enter). En este punto nosotros habremos ingresado en el programa tanto el entero positivo como los dos caracteres `/n` correspondientes al salto de línea.

El `scanf` se queda con el número ingresado, omitiendo la existencia de los caracteres `/n`, los cuales se "empujan" hacia la siguiente sentencia donde se encuentra el `gets`. Esta función en vez de esperar a que nosotros escribamos la cadena de caracteres, toma el `/n` que se arrastró hasta ese punto y lo almacena en el arreglo de caracteres "arreglo[10]".

Si pudiésemos ver en memoria lo que acabamos de realizar, el casillero "entero" tendría un número entero positivo (el que nosotros ingresamos por teclado), mientras que los casilleros correspondientes al arreglo de caracteres tendrían escrito `/n`.

```
main(){
int entero;
char arreglo[10];

printf("Ingrese un entero positivo: ");
scanf("%d",&amp;entero);
fflush(stdin);
printf("\nIngrese una cadena de caracteres: ");
gets(arreglo);

printf("\n\nUsted ingreso %d como Entero y %s como cadena de
caracteres\n",entero,arreglo);
system("pause");
}
```

Este último código muestra como se puede evitar el inconveniente provocado de utilizar el `gets` luego del `scanf` mediante la función `fflush(stdin);` que se encarga de vaciar el buffer, o sea de desechar el `/n` luego de la ejecución del `scanf`, para que de esta forma el programa no se saltee el `gets` y tengamos nosotros que escribir la cadena de caracteres a almacenar en el arreglo de caracteres.

Este artículo lo escribí en base a fflush vs gets publicado por Steve Summit, el cual se encuentra en Ingles y abarca lo mismo tratado aquí pero de forma más compleja.