Universidad Centroamericana “José Simeón Cañas”
Facultad de Ingeniería y Arquitectura

Tutorías - Fundamentos de Programación Ciclo 01/2021
Tutor: Ernesto José Canales Guillén

Introducción a la programación en C++

Flujos de E/S y dispositivos de E/S estándar

Output

En C ++, la salida en el dispositivo de salida estándar se logra mediante el uso de *cout* y el operador <<. La sintaxis general de *cout* junto con << es:

```
cout << expression or manipulator << expression or manipulator...;
```

Esto se llama declaración de salida. En C ++, << se llama inserción de flujo operador. La generación de resultados con cout sigue dos reglas:

1. La expresión se evalúa y su valor se imprime en el valor actual. punto de inserción en el dispositivo de salida.
2. Se utiliza un manipulador para formatear la salida. El manipulador más simple es endl (el último carácter es la letra el), lo que provoca la inserción apunte para moverse al principio de la siguiente línea.

Secuencias de escape de uso común:

En ciertas ocasiones, necesitaremos escapar caracteres/valores o necesitaremos dar un leve formato al texto; para escapar carácter se hace uso de la pleca invertida (\):

	Escape Sequence	Description
`\n`	Newline	Cursor moves to the beginning of the next line
`\t`	Tab	Cursor moves to the next tab stop
`\b`	Backspace	Cursor moves one space to the left
`\r`	Return	Cursor moves to the beginning of the current line (not the next line)
`\\`	Backslash	Backslash is printed
`\'`	Single quotation	Single quotation mark is printed
`\"`	Double quotation	Double quotation mark is printed

Input

Cuando la computadora obtiene los datos del teclado, se dice que el usuario actúa de forma interactiva. La colocación de datos en variables desde el dispositivo de entrada estándar se logra mediante el uso de *cin* y el operador >>. La sintaxis de *cin* junto con >> es:

```
cin >> variable >> variable ...;
```

Esto se denomina declaración de entrada (lectura). En C ++, >> se llama extracción de flujo operador.

cin nos permite usar funciones predefinidas, tales como *get*, *ignore*, *peek* y *putback*; para ingresar los datos de entrada de una manera específica.

cin y la función *get*

La variable cin puede acceder a la función de flujo get, que se usa para leer caracteres de datos. La función get ingresa el siguiente carácter, incluidos los espacios en blanco, del flujo de entrada y lo almacena en la ubicación de memoria indicada por su argumento. La función get viene de muchas formas.

```
cin.get(varChar);
```

cin y la función *ignore*

Cuando desee procesar solo datos parciales (por ejemplo, dentro de una línea), puede usar la secuencia función ignorar para descartar una parte de la entrada.

```
cin.ignore(intExp, chExp);
```

Aquí, intExp es una expresión entera que produce un valor entero y chExp es un char expresión que produce un valor char. De hecho, el valor de la expresión intExp especifica el número máximo de caracteres a ignorar en una línea

Las funciones de *putback* y *peek*

La función *putback* le permite poner el último carácter extraído de la entrada por la función de get de nuevo en el flujo de entrada. La función de flujo mira en el flujo de entrada y le dice cuál es el siguiente carácter sin eliminarlo del flujo de entrada. Al usar estas funciones, después de determinar que la siguiente entrada es un número, puede leerlo como un número. No es necesario que lea los dígitos del número como caracteres y luego convierta estos caracteres a ese número.

```
istreamVar.putback(ch);
```

Aquí, istreamVar es una variable de flujo de entrada, como cin, y ch es una variable char.

La función *peek* devuelve el siguiente carácter del flujo de entrada, pero no elimina el caracter. En otras palabras, la función peek busca en el flujo de entrada y comprueba la identidad del siguiente carácter de entrada. Además, después de comprobar la siguiente entrada carácter en el flujo de entrada, puede almacenar este carácter en una ubicación de memoria designada sin eliminarlo del flujo de entrada. Es decir, cuando usa la función peek, el siguiente carácter de entrada permanece igual, aunque ahora ya sepa cuál es.

```
ch = istreamVar.peek();
```

Aquí, istreamVar es una variable de flujo de entrada, como cin, y ch es una variable char.

La función *clear*

Cuando un flujo de entrada entra en el estado de falla, el sistema ignora todas las E / S adicionales. Puede utilizar la función *clear* para restaurar el flujo de entrada a un funcionamiento expresar.

```
istreamVar.clear();
```

Después de usar la función clear para devolver el flujo de entrada a un estado de trabajo, todavía necesita borrar el resto de la basura del flujo de entrada. Esto se puede lograr usando la función ignore.

Input/Output y el tipo de *string*

Recuerde que el operador de extracción omite los espacios en blanco iniciales y que la lectura se detiene en un carácter de espacio en blanco. Como consecuencia, no puede utilizar la extracción operador para leer cadenas que contienen espacios en blanco.

Para leer una cadena que contiene espacios en blanco, puede usar la función *getline*.

```
getline(istreamVar, strVar);
```

Donde istreamVar es una variable de flujo de entrada y strVar es una variable de cadena. La lectura está delimitada por el carácter de nueva línea '\ n'. La función getline lee hasta que llega al final de la línea actual. La nueva línea del carácter también se lee, pero no se almacena en la variable de cadena.