Lecture 7 Program Statement (English)

Many lines in C programs are considered program statements, which serve to control program execution and functionality. Many of these program statements must end with a statement terminator. Statement terminators are simply semicolons (;). The next line of code, which includes the printf() function, demonstrates a program statement with a statement terminator.

```
printf("\nC you later\n");
```

Some common program statements that do not require the use of statement terminators are the following:

• Comments

• Preprocessor directives (for example, #include or #define)

• Begin and end program block identifiers

• Function definition beginnings (for example, main())

The preceding program statements don't require the semicolon (;) terminator because they are not executable C statements or function calls. Only C statements that perform work during program execution require the semicolons.

A function commonly used for displaying output to the computer screen is the printf() function. As shown next, the printf() function is used to write the text "C you later" to the standard output.

```
printf("\nC you later\n");
```

Like most functions, the printf() function takes a value as a parameter. (I'll talk more about functions in Chapter 5, "Structured Programming.") Any text you want to display in the standard output must be enclosed by quotation marks.

For the most part, characters or text that you want to appear on-screen are put inside quotation marks, with the exception of escape characters or escape sequences. The backslash character (\) is the escape character. When the printf() statement shown above is executed, the program looks forward to the next character that follows the backslash. In this case, the next character is the character n. Together, the backslash (\) and n characters make up an escape sequence.

Escape Sequences

Escape sequences are specially sequenced characters used to format output.

This particular escape sequence (\n) tells the program to add a new line. Take a look at the following program statement. How many new lines are added to standard output with this one printf() function?

```
printf("\nC you later\n");
```

This printf() function adds two new lines for formatting purposes. Before any text is shown, the program outputs a new line. After the text is written to standard output, in this case the computer screen, another new line is written.

some common escape sequences.

TABLE 1.2 COMMON ESCAPE SEQUENCES

Escape Sequence	Purpose
\n	Creates a new line
\t	Moves the cursor to the next tab
\r	Moves the cursor to the beginning of the current line
\\	Inserts a backslash
\"	Inserts a double quote
\'	Inserts a single quote

Lecture 7 Program Statement (Hindi)

सी प्रोग्राम्स में कई पंक्तियाँ प्रोग्राम स्टेटमेंट मानी जाती हैं, जो प्रोग्राम निष्पादन और कार्यक्षमता को नियंत्रित करने के लिए सेवा प्रदान करती हैं। इनमें से कई प्रोग्राम स्टेटमेंट को स्टेटमेंट टर्मिनेटर के साथ समाप्त करना होता है। स्टेटमेंट टर्मिनेटर सिर्फ अर्धविराम (;) होते हैं। निम्नलिखित कोड की पंक्ति, जिसमें printf() फ़ंक्शन शामिल है, एक प्रोग्राम स्टेटमेंट को दिखाती है जिसमें स्टेटमेंट टर्मिनेटर होता है।

printf("\nC आप से बाद में मिलेंगे\n");

कुछ आम प्रोग्राम स्टेटमेंट्स जिनके लिए स्टेटमेंट टर्मिनेटर्स की आवश्यकता नहीं होती है, निम्नलिखित हैं:

• टिप्पणियाँ

• प्रीप्रोसेसर निर्देश (उदाहरण के लिए, #include या #define)

• प्रोग्राम ब्लॉक पहचानकर्ताओं की शुरुआत और अंत

• फ़ंक्शन डेफ़िनिशन की शुरुआत (उदाहरण के लिए, main())

पूर्वोक्त प्रोग्राम स्टेटमेंट्स को अर्धविराम (;) टर्मिनेटर की आवश्यकता नहीं होती क्योंकि वे निष्पादित सी स्टेटमेंट्स या फ़ंक्शन कॉल्स नहीं हैं। केवल वे सी स्टेटमेंट्स जो प्रोग्राम निष्पादन के दौरान काम करते हैं, अर्धविरामों की आवश्यकता होती है।

कंप्यूटर स्क्रीन पर आउटपुट दिखाने के लिए आमतौर पर उपयोग किया जाने वाला एक फ़ंक्शन printf() फ़ंक्शन है। जैसा कि अगले में दिखाया गया है, printf() फ़ंक्शन का उपयोग मानक आउटपुट में टेक्स्ट “C आप से बाद में मिलेंगे” लिखने के लिए किया जाता है।

```
printf("\nC आप से बाद में मिलेंगे\n");
```

अधिकांश फ़ंक्शनों की तरह, printf() फ़ंक्शन एक मान को पैरामीटर के रूप में लेता है। (मैं अध्याय 5, “संरचित प्रोग्रामिंग” में फ़ंक्शनों के बारे में और बात करूँगा।) जो भी टेक्स्ट आप मानक आउटपुट में प्रदर्शित करना चाहते हैं, उसे उद्धरण चिह्नों के बीच में रखना होगा।

ज्यादातर मामलों में, जो अक्षर या टेक्स्ट आप स्क्रीन पर दिखाना चाहते हैं, उन्हें उद्धरण चिह्नों के अंदर रखा जाता है, बच

के अक्षर या एस्केप अनुक्रमों को छोड़कर। बैकस्लैश अक्षर (\) एस्केप अक्षर है। जब उपरोक्त दिखाया गया printf() स्टेटमेंट निष्पादित किया जाता है, तो प्रोग्राम बैकस्लैश के बाद आने वाले अगले अक्षर की ओर देखता है। इस मामले में, अगला अक्षर n अक्षर है। साथ में, बैकस्लैश (\) और n अक्षर एक एस्केप अनुक्रम बनाते हैं।

Escape Sequence

एस्केप सीक्वेंसेस विशेष रूप से सेट की गई पंक्तियों होती हैं जो आउटपुट को स्वरूपित करने के लिए उपयोग किए जाते हैं।

यह विशेष एस्केप सीक्वेंस (\n) प्रोग्राम को नई पंक्ति जोड़ने का निर्देश देता है। निम्नलिखित प्रोग्राम स्टेटमेंट पर एक नज़र डालें। इस एक printf() फंक्शन के साथ मानक आउटपुट में कितनी नई पंक्तियाँ जोड़ी जाती हैं?

printf("\nC आप से बाद में मिलेंगे\n");

यह printf() फंक्शन स्वरूपण उद्देश्यों के लिए दो नई पंक्तियाँ जोड़ता है। किसी भी पाठ को दिखाने से पहले, प्रोग्राम एक नई पंक्ति आउटपुट करता है। पाठ को मानक आउटपुट पर लिखने के बाद, इस मामले में कंप्यूटर स्क्रीन पर, एक और नई पंक्ति लिखी जाती है।

कुछ सामान्य एस्केप सीक्वेंसेस।

TABLE 1.2 COMMON ESCAPE SEQUENCES

Escape Sequence	Purpose
\n	Creates a new line
\t	Moves the cursor to the next tab
\r	Moves the cursor to the beginning of the current line
\\	Inserts a backslash
\"	Inserts a double quote
\'	Inserts a single quote