فصل سوم

ثابتها

ثابتها (Constants) عبارتهایی با مقدار ثابت و قطعی هستند.

لیترال ها

لیترالها (literals) برای بیان مقادیر مشخص در متن کد (source code) مورد استفاده قرار می‌گیرند. در فصل قبل ما از لیترالها برای قرار دادن مقدار، درون متغیر یا بیان یک عبارت جهت چاپ استفاده کردیم. مثال:

```
a = ٥;
```

در این قطعه کد عدد ۵ یک لیترال می باشد. ثابتهای لیترالی می توانند اعداد صحیح، اعداد اعشاری، کاراکتر، رشته های حرفی، و مقادیر منطقی باشند.

اعداد صحیح

```
١٧٧٦
٧٠٧
-٢٧٣
```

اینها ثابتهای عددی هستند که مقادیر صحیح در مبنای ۱۰ را نشان می دهند. برای نوشتن ثابتهای عددی نیازی به استفاده از " و یا هر کاراکتر دیگری نیست. علاوه بر مبنای ۱۰ در C++ می توان از مبنای ۸ و یا ۱۶ نیز استفاده کرد. به این ترتیب که جهت استفاده از مبنای ۸ ابتدای عدد، رقم صفر (۰) و برای استفاده از مبنای ۱۶ در ابتدای عدد صفر و ایکس (۰x) قرار می گیرد. مقادیر زیر همگی با هم برابر هستند:

```
٧٥          // decimal
٠١١٣        // octal
٠x٤b        // hexadecimal
```

تمام اعداد بالا بیانگر عدد ۷۵ هستند. اولی در مبنای ۱۰، دومی در مبنای ۸ و سومی در مبنای ۱۶ می باشد.

کامپایلر همواره به لیترالها یک نوع را اختصاص می‌دهند. لیترالهای عددی صحیح به عنوان `int` در نظر گرفته می شوند. به علاوه ما می توانیم صریحا مشخص کنیم که لیترال از نوع `unsigned int` و یا `long int` است؛ به این صورت که در انتهای عدد اگر `u` قرار گیرد بیانگر `unsigned int` و اگر `l` قرار گیرد بیانگر `long int` می‌باشد. مثال:

```
٧٦          // int
٧٦u         // unsigned int
٧٦l         // long
٧٦ul        // unsigned long
```

پسوندهای بالا هم می توانند حروف بزرگ باشند و هم حروف کوچک.

اعداد ممیز شناور (اعشاری)

این اعداد از یک عدد دهدهی به همراه e و یک عدد صحیح تشکیل شده اند. عددی که بعد از حرف e می آید بیانگر توان ۱۰ عدد اعشاری می باشد. مثال:

```
۳,۱٤۱٥۹     // ۳,۱٤۱٥۹
٦,۰۲e۲۳     // ٦,۰۲ x ۱۰۲۳
۱,٦e-۱۹     // ۱,٦ x ۱۰-۱۹
۳,۰         // ۳,۰
```

اولین عدد بیانگر عدد پی (Л)، دومی عدد آوگادرو و سومی مقدار شارژ الکتریکی یک الکترون است. تمام این اعداد با تقریب محاسبه شده اند هستند.

نوع پیش فرض برای لیترالهای اعشاری double می باشد. اگر شما صریحا بخواهید نوع آن را مشخص کنید برای float از f و برای double از l در انتهای عدد استفاده می کنیم. مثال:

```
۳,۱٤۱٥۹L   // long double
٦,۰۲e۲۳f   // float
```

حروف e، f و l که در اعداد اعشاری ثابت استفاده می شوند هم می توانند بزرگ و هم می توانند کوچک باشند و تفاوتی میان حالت این حروف وجود ندارد.

کاراکترها و رشته ها

کاراکترها و رشته ها ثابتهای غیر عددی هستند که از مجموعه‌ای از یک یا چند کاراکتر تشکیل شده‌اند. مثال:

```
'z'
'p'
"Hello world"
"How do you do?"
```

دو عبارت اول بیانگر ثابت تک کاراکتری هستند و دو عبارت بعدی رشته های حرفی را تشکیل می دهند. توجه داشته باشید برای بیان یک کاراکتر آن را بین (') و برای بیان رشته ها (که بیش از یک کاراکتر هستند) آن را بین (") قرار می‌دهند. هنگام استفاده از کاراکتر و یا رشته حتماً می‌بایست آنها را بین ' و " قرار داد تا کامپایلر آنها را با متغیر اشتباه نگیرد. به عنوان مثال دو عبارت زیر با هم کاملا تفاوت دارند:

```
x
'x'
```

x بیانگر یک متغیر با نام x می‌باشد در حالی که 'x' بیانگر یک لیترال کاراکتری می‌باشد.

در رشته ها کاراکترهایی وجود دارد که نمایش آنها مشکل و یا غیر ممکن است مانند کاراکتر خط جدید و یا کاراکتر tab. برای نمایش این کاراکترها از \ به همراه یک حرف استفاده می شود به این کاراکترها back slash characters هم گفته می‌شود. در زیر لیستی از این کاراکترها آورده شده است:

\n	newline	خط جدید

\r	carriage return	به ابتدای سطر رفتن
\t	tab	۸ کاراکتر به جلو
\v	vertical tab	۸ خط به پایین
\b	backspace	پاک کردن کاراکتر قبلی
\f	form feed (page feed)	صفحه بعد
\a	alert (beep)	بوق اخطار
\'	single quote (')	کاراکتر '
\"	double quote (")	کاراکتر "
\?	question mark (?)	علامت سوال
\\	backslash (\)	کاراکتر \

مثال:

```
'\n'
'\t'
"Left \t Right"
"one\ntwo\nthree"
```

به علاوه شما می‌توانید هر کاراکتری را تنها با آوردن کد ASCII آن کاراکتر بعد از \ بیان کنید. عدد کد اسکی می‌بایست در مبنای ۸ و یا در مبنای ۱۶ باشد. در حالت اول (مبنای ۸) ارقام درست بعد از \ قرار می‌گیرند. اما در حالت دوم بعد از \ حرف x و سپس ارقام در مبنای ۱۶ قرار می‌گیرند.

لیترالهای رشته‌ای می‌توانند در بیش از یک خط امتداد پیدا کنند این کار با قرار دادن \ در انتهای خط ناتمام انجام می‌پذیرد. مثال:

```
"string expressed in \
two lines"
```

شما حتی می‌توانید چند رشته را به یکدیگر اتصال دهید. مثال:

```
"this forms" "a single" "string" "of characters"
```

در بین رشته‌ها می‌توان از فضای خالی، خط جدید، + و دیگر کاراکترهای مجاز که فضای خالی را بیان می‌کنند استفاده نمود.

لیترالهای منطقی

تنها دو مقدار معتبر برای این نوع وجود دارد: true و false. در C++ نوع این مقادیر به عنوان bool در نظر گرفته می‌شود.

ثابتهای تعریف شده (#define)

شما با استفاده از رهنمون پیش پردازنده #define می‌توانید ثابتهایی را تعریف کرده و یک مقدار به آن نسبت دهید. روش تعریف این ثابتها به شکل زیر است:

```
#define identifier value
```

مثال:

```
#define PI ۳,۱٤۱۵۹۲٦۵
#define NEWLINE '\n'
```

عبارات بالا دو ثابت جدید به نامهای PI و NEWLINE تعریف می‌کنند. وقتی شما این ثابتها را تعریف کردید می‌توانید در بقیه کد از آنها همانند دیگر لیترالها استفاده کنید. مثال:

```
// defined constants: calculate circumference

#include <iostream>
using namespace std;

#define PI ۳,۱٤۱۵۹
#define NEWLINE '\n';

int main ()
{
  double r=۵,۰;                // radius
  double circle;

  circle = ۲ * PI * r;
  cout << circle;
  cout << NEWLINE;

  return ۰;
}
```

نتیجه:

```
۳۱,٤۱۵۹
```

کاری که پیش‌پردازنده در اینجا انجام می‌دهد این است که هر جا درون کد، متغیر تعریف شده بعد از # (در این مثال PI و NEWLINE) وجود داشته باشد، آن را با مقدار تعیین شده در خط رهنمون تعویض می‌کند.

دستور #define یک عبارت معتبر در c++ نیست ولی به عنوان یک دستور تعریف شده در پیش‌پردازنده به کار می‌رود، در نتیجه نیازی به قرار دادن ; در انتهای جمله نمی‌باشد. اگر شما از ; در انتهای جمله استفاده کنید آنوقت ; در هر جایی که پیش‌پردازنده عملیات تعویض را انجام می‌دهد اضافه خواهد شد.

ثابتهای اعلان شده (const)

با قرار دادن پیشوند const در هنگام تعریف یک متغیر شما می‌توانید یک ثابت با نوع مشخص همانند یک متغیر تعریف کنید. مثال:

```
const int pathwidth = ١٠٠;
const char tabulator = '\t';
const zipcode = ١٢٤٤٠;
```

در خط سوم چون ما نوع متغیر را تعریف نکرده‌ایم، کامپایلر نوع این متغیر را به طور پیش فرض int در نظر می‌گیرد.