Assignment, Formatting, and Interactive Input
Assignment

Variable = expression

• A variable is an identifier

• An expression can be:
  • A constant:
    - radius = 2.5,
    - height = 16
  • an arithmetic statement:
    - volume = 3.146 * radius * radius * height;
```cpp
#include <iostream>

using namespace std;

int main()
{
    double radius, height, volume;
    radius = 2.5;
    height = 16;
    volume = 3.1416 * radius * radius * height;
    cout << "The volume of the cylinder is " << volume << endl;
    return 0;
}
```

Output: **The volume of the cylinder is 314.599**
expressions

cout << "The value is " << (2+8+12);
The value is 22

cout << "The value is " << (a=5);
The value is 5;

• expressions get evaluated as soon as the compiler sees them (eager evaluation)

• expressions get stored as the values they are evaluated to
overwriting values

```cpp
sum = 25;
cout << "sum = " << sum << endl;

sum = 25

sum = sum + 10;
cout << "sum = " << sum;

sum = 35
```
Accumulators

- used to keep track of subtotals
- operators: \( +=, -=, *=, /=, %= \)

\textit{sum} += 10 \textit{is the same as} \textit{sum} = \textit{sum} + 10 \\
\textit{sum} = 10 + 20 + 30

<table>
<thead>
<tr>
<th>statement</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textit{sum} = 0</td>
<td>0</td>
</tr>
<tr>
<td>\textit{sum}+=10</td>
<td>10</td>
</tr>
<tr>
<td>\textit{sum}+=20</td>
<td>30</td>
</tr>
<tr>
<td>\textit{sum}+=30</td>
<td>60</td>
</tr>
</tbody>
</table>
## counters

<table>
<thead>
<tr>
<th>name</th>
<th>statement</th>
<th>equivalence</th>
</tr>
</thead>
</table>
| prefix   | $k = ++n;$  | $n = n + 1;$
|          |             | $k = n;$                             |
| postfix  | $k = n++;$  | $k = n;$                             |
|          |             | $n = n + 1;$                         |

- counters: variable += fixedNumber
- variable +=1 <=> variable++/++variable
- variable -=1 <=> variable--/--variable
### Formatting Numbers

<table>
<thead>
<tr>
<th>no formatting</th>
<th>formatting</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>18</td>
<td>18</td>
</tr>
<tr>
<td>124</td>
<td>124</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>148</td>
<td>148</td>
</tr>
</tbody>
</table>

- `cout<<setw(3) << 6 << endl;`
- `setw(3)` creates a field with of 3
- `cout<<”---\n”` prints out `---` and a new line
- `#include<iomanip>`
Format flags

- flag – sets things active or non-active (T/F)
- flags in iosmanip:
  - set output as exponential or decidable
  - right or left justify the output
  - show a + for positive numbers
- functions also exist to do all these things
  - flags set format until flag state is changed
  - functions only set format for expression they're part of
Formatting cout

cout << setiosflags(ios::fixed)

<=>

cout.setf(ios::fixed)

- setiosflags and setf take all the same arguments (table 3.3)
- also works for setprecision()/ cout.precision() and setfill()/cout.fill()
- can also set justification and endl in setf
Using math functions

math equivalent:

\[ f(x) = y \]
input: \( x \)
output: \( y \)
cylinder program using cmath lib

- add: #include <cmath> at the top
- change
  \[
  \text{volume} = 3.146 \times \text{radius} \times \text{radius} \times \text{height};
  \]
  to
  \[
  \text{volume} = 3.146 \times \text{pow} (\text{radius}, 2) \times \text{height}
  \]

better way to calculate cylinder

```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main()
{
    const double PI = 3.1416;
    double radius, height, volume;
    radius = 2.5;
    height = 16;
    volume = PI * pow(radius, 2) * height;
    cout << "The volume of the cylinder is " << volume << endl;
    return 0;
}
```
order of evaluation

1) 2.0 * sqrt(5*10-25) / 2.0
2) 2.0 * sqrt(50-25) / 2.0
3) 2.0 * sqrt(25) / 2.0
4) 2.0 * 5/2.0
5) 10.0/2.0
6) 5.0
cin

- `cin >> variable;`
- `<<` means user input gets put into variable
- Part of the iostream library, so you need to write
  ```
  #include<iostream>
  ```
- iostream uses the std namespace, so write:
  ```
  using namespace std;
  ```
- otherwise you need to specify namespace
  ```
  std::cin >> variable;
  ```
#include <iostream>
using namespace std;

int main()
{
    int day, month, year;
    cout << “What year were you born in? ”;
    cin >> year;
    cout <<” And in which month? “;
    cin >> month;
    cout << “And what day?”; 
    cin >> day;
    cout <<”You were born on “<<month<<”/”<<day<<”/”<<year;
    return 0;
}
#include <iostream>
using namespace std;

int main()
{
int day, month, year;
cout << "What's your dob? (enter in MM DD YYYY form) ";
cin >> month >> day >> year;
cout <<"You were born on "<<month<<"/"<<day<<"/"<<year;
return 0;
}
<table>
<thead>
<tr>
<th>Old</th>
<th>New</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>radius = 2.5;</code></td>
<td><code>cout &lt;&lt; &quot;This program calculates the volume of a cylinder&quot; &lt;&lt; endl;</code></td>
</tr>
<tr>
<td><code>height = 16;</code></td>
<td><code>cout &lt;&lt; &quot;What's the radius of the cylinder? \n&quot;;</code></td>
</tr>
<tr>
<td></td>
<td><code>cin &gt;&gt; radius;</code></td>
</tr>
<tr>
<td></td>
<td><code>cout &lt;&lt; &quot;What's the height of the cylinder?&quot;</code></td>
</tr>
<tr>
<td></td>
<td><code>cin &gt;&gt; height;</code></td>
</tr>
</tbody>
</table>
```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main()
{
    const double PI = 3.1416;
    double radius, height, volume;
    cout << "This program calculates the volume of a cylinder" << endl;
    cout << "What's the radius of the cylinder? ";
    cin >> radius;
    cout << "What's the height of the cylinder? ";
    cin >> height;
    volume = PI * pow(radius, 2) * height;
    cout << "The volume of the cylinder is " << volume << endl;
    return 0;
}
```
user-input validations

● some values don't make any sense
  ● for dob:
    - days only go between 1-31 (not even for some months)
    - months only go between 1-12
    - unlikely to ever run into someone born before 1900

● methods for checking values covered in later chapters
Quiz

• How do I store the contents of x+y in z?
  • a) z = x+y;
  • b) z = x;
    \[z += y;\]
  • c) z = y;
    \[z += x;\]
  • d) all of the above
HW

• Quiz prep: Write a program to:
  • obtain the radius of a sphere from the user
  • calculate the area of the sphere
  • output answers in scientific notation.

• reading: chapter 3