

CMPS 150 Workbook: Chapter 1 and Strings

Section 1.2: Variables, Expressions and Assignment Statements

1) Which are the correct identifiers:

`_LAMA` `total` `1984` `sum Grades`

2) Which are the correct identifiers:

`1a` `a1` `horse at_rest` `year2020`

3) Which data types fit the following literal values?

- a) 13
- b) 13.0
- c) 'a'
- d) false
- e) -123.9

4) Which data types fit the following literal values?

- a) true
- b) -4
- c) 17.6
- d) 6003
- e) '\$'

5) Using the following identifiers, declare variables of the given data types.

- a) Identifier: `year`, Data Type: `int`
- b) Identifier: `totalPay`, Data Type: `double`
- c) Identifier: `choice`, Data Type: `char`
- d) Identifier: `isActive`, Data Type: `bool`

6) Using the following identifiers, declare variables of the given data types.

- a) Identifier: `seat`, Data Type: `int`
- b) Identifier: `netCost`, Data Type: `double`
- c) Identifier: `guess`, Data Type: `char`
- d) Identifier: `notYet`, Data Type: `bool`

7) Using the following variable identifiers, assign the variable identifiers the given values:

- a) Variable identifier: year, Value to assign: 2011
- b) Variable identifier: totalPay, Value to assign: 3042.23
- c) Variable identifier: choice, Value to assign: '1'
- d) Variable identifier: isActive, Value to assign: false
- e) Variable identifiers: a, b; Value to assign: 0

8) Using the following variable identifiers, assign the variable identifiers the given values:

- a) Variable identifier: seat, Value to assign: 45
- b) Variable identifier: netCost, Value to assign: -15.8
- c) Variable identifier: guess, Value to assign: '&'
- d) Variable identifier: notYet, Value to assign: true

9) Declare an appropriate variable identifier of an appropriate type and assign it the indicated value using only one statement.

- a) Declare a variable to hold a high temperature.
Value: 89.5
- b) Declare a variable to hold the number of people admitted to view a movie.
Value: 231

10) Declare an appropriate variable identifier of an appropriate type and assign it the indicated value using only one statement.

- a) Declare a variable to hold the day of the month
Value: 14
- b) Declare a variable to hold the balance of a bank account.
Value: 34.78

11) Evaluate the following expressions and give the results.

- a) $3 + 4 * 5 - 4$
- b) $(3 + 4) * (5 - 4)$
- c) $2.7 - 10.0 / 4.0$
- d) $7 / 3$
- e) $7 \% 3$
- f) $7.0 / 3.0$

12) Evaluate the following expressions and give the results.

- a) $8 - 4 / 2 + 1$
- b) $(8 - 4) / 2 + 1$
- c) $6.1 + 12.0 * 2.5$
- d) $2.7 - 10.0 / 4.0$
- e) $12 / 5$
- f) $12 \% 5$
- g) $12.0 / 5.0$

13) Given that the variable identifier *length* has the value 3 and the variable identifier *width* has the value 5, what is the value of the variable identifier *area* after the following assignment statement?

```
area = length * width;
```

14) Given that the variable identifier *weight* has the value 35 and the variable identifier *quantity* has the value 3, what is the value of the variable identifier *grossWeight* after the following assignment statement?

```
grossWeight = weight * quantity;
```

15) Given that the variable identifier *count* has the value of 7 before the following statement, what is the value of *count* after the following statement.

```
count++;
```

16) Given that the variable identifier *count* has the value of 7 before the following statement, what is the value of *count* after the following statement.

```
count--;
```

17) Using the following identifiers, declare named constants of the given name using appropriate types and assign them the given values:

a) constant identifier: *PI*, Value to assign: 3.14159

b) constant identifier: *MAX*, Value to assign: 100

18) Using the following identifiers, declare named constants of the given name using appropriate types and assign them the given values:

a) constant identifier: *X_LIMIT*, Value to assign: 648

b) constant identifier: *MAX_WEIGHT*, Value to assign: 3042.23

Section 1.3: Console Input / Output

1) What is the output of the following code?

```
cout << "Of all human ills, " << endl
     << "greatest is fortune\'s\nwayward tyranny.";
```

2) What is the output of the following code?

```
cout << "Whom the gods\nwould destroy, " << endl
     << "they first make mad.";
```

3) What is the output of the following code?

```
double price = 100.0;
double tax = .08;
cout << "total cost: $" << price * tax + price;
```

4) What is the output of the following code?

```
int width = 3;
int length = 15;
int area;
area = width * length;
cout << "The area of a rectangle that is " << width
     << " by " << length << " feet is " << area
     << " square feet.";
```

5) What is the output of the following code?

```
cout.setf(ios::fixed, ios::floatfield);
cout.setf(ios::showpoint);
cout.precision(2);
double price = 100.0;
double tax = .085;
cout << "total cost: $ " << price * tax + price;
```

6) What is the output of the following code?

```
const double PI = 3.1459;
int p = 3;
cout.setf(ios::fixed, ios::floatfield);
cout.setf(ios::showpoint);
cout.precision(p);
cout << "PI to " << p << " places is " << PI;
```

7) Write the code to produce the following output.

```
Talk sense
to a fool
and he calls you foolish.
```

8) Write the code to produce the following output.

```
There's no point
in being grown up
if you can't be childish
sometimes.
```

9) Assuming that the user enters 3 when prompted by the program, what is the output of the following code?

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

int main()
{
    cout.setf(ios::fixed, ios::floatfield);
    cout.setf(ios::showpoint);
    cout.precision(2);

    double apples;

    cout << "How many apples do you want? ";
    cin >> apples;
    cout << "That will be $" << apples * .45 << endl;

    return 0;
}
```

10) Assuming that the user enters 1.5 and 3 when prompted by the program, what is the output of the following code?

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

const double PI = 3.14159;

int main()
{
    cout.setf(ios::fixed, ios::floatfield);
    cout.setf(ios::showpoint);
    cout.precision(2);

    double width, length;

    cout << "Enter the length in feet: ";
    cin >> length;
    cout << "Enter the width in feet: ";
    cin >> width;

    cout << "The area is " << length * width
         << " square feet." << endl;

    return 0;
}
```

11) Write a complete C++ program to calculate the area of a circle expressed in feet. (area = πr^2)

12) Write a complete C++ program to calculate the average of three test scores.

Section 9.3: Standard String Class

1) What is the output of the following code?

```
string part1 = "Reason and judgment ";
string part2 = "are the qualities ";
string part3 = "of a leader.";

cout << part1 << part2 << part3 << endl << "-Tacticus";
```

2) What is the output of the following code?

```
string aLine = "Only the educated ";
string bLine = "-Epictetus";

cout << aLine << "are free " << bLine;
```

3) What is the output of the following code?

```
string word = "generous";
string proverb = "No one is so ";

proverb = proverb + word
          + " as he who has nothing to give.";
cout << proverb;
```

4) What is the output of the following code?

```
string word1 = "quarrel";
string word2 = "argue";
string phrase = "because they cannot";
string quote;

quote = "People generally " + word1 + " " + phrase
        + " " + word2;
cout << quote << endl << "-Gilbert K. Chesterton";
```

5) If the user enters the string

```
one two
```

when the following program is run, what is the output?

```
#include <iostream>
#include <string>
using namespace std;

int main()
{
    string oneString;

    cout << "Enter something: ";
    cin >> oneString;
    cout << oneString << endl;

    return 0;
}
```

6) If the user enters the strings

```
one           // first prompt
three four    // second prompt
```

when the following program is run, what is the output?

```
#include <iostream>
#include <string>
using namespace std;

int main()
{
    string oneString, twoString;

    cout << "Enter something: ";
    cin >> oneString;
    cout << "Enter something else: ";
    cin >> twoString;
    cout << oneString << ' ' << twoString << endl;

    return 0;
}
```

7) If the user enters the string

```
one two
```

when the following program is run, what is the output?

```
#include <iostream>
#include <string>
using namespace std;

int main()
{
    string oneString;

    cout << "Enter something: ";
    getline(cin, oneString);
    cout << oneString << endl;

    return 0;
}
```

8) If the user enters the strings

```
one two          // first prompt
three four       // second prompt
```

when the following program is run, what is the output?

```
#include <iostream>
#include <string>
using namespace std;

int main()
{
    string oneString, twoString;

    cout << "Enter something: ";
    getline(cin, oneString);
    cout << "Enter something else: ";
    getline(cin, twoString);
    cout << oneString << ' ' << twoString << endl;

    return 0;
}
```

More on Formatting Output

The *ostream* object *cout* handles output of all basic data types. There are several functions that belong to *ostream* class that are part of *cout*. For example, using *setf* as follows

```
cout.setf(ios::fixed, ios::floatfield);  
cout.setf(ios::showpoint);
```

directs *cout* to output all floating point values in decimal format. By further using *precision*, the number of digits to the right of the decimal point can be set.

```
cout.precision(#);
```

where # is an integer that is 0 or greater.

Function *precision* can be used as often as necessary to alter the number of decimal places. Functions *setf* and *precision* are likely already familiar to you.

Two other *ostream* class functions are worth mentioning now. These are *width*, which specifies the minimum number of places the next output will occupy, and *fill*, which specifies the character that will be used as the filler character. The syntax for these functions is as follows:

```
cout.width(#);
```

where # is an integer expression that has a value of 0 or greater

```
cout.fill(char);
```

where char is a char expression, example: 'a'

Function *width* operates on all data types except *char*. (Note that since class *string* is not a basic data type, these functions will not affect output from class *string* variables or constants. Ordinary strings of quoted characters will be affected by these functions.) Here are some examples of using *width* and *fill*.

Example 1:

```

string name = "Fred", home = "Mars";
int age = 34;
cout << "123456789 123456789 123456789" << endl;
cout.width(20);
cout << "My name is: " << name << endl;
cout.width(20);
cout << "I am: " << age << endl;
cout.width(20);
cout << "I was born on: " << home << endl;

// Output for example 1 is as follows:

```

```

123456789 123456789 123456789
    My name is: Fred
        I am: 34
    I was born on: Mars

```

Notice that the string at the start of each `cout` takes up 20 spaces and is right justified. This has the effect of lining up the colon-space at the end of each beginning string. Notice that *width* only effects the next output value.

Example 2:

```

cout.setf(ios::fixed, ios::floatfield);
cout.setf(ios::showpoint);
double hours=33.5, rate=12.75;
cout.precision(2);
cout << "Gross pay: $";
cout.fill('*');
cout.width(10);
cout << hours * rate << endl;
cout.fill(' ');

// Output for example 1 is as follows:

```

```

Gross pay: $*****427.12

```

Notice the use of *fill* to set the fill character to the asterisk in order to pad the monetary amount. (This is a safety practice often used when printing checks.) (Notice also that *fill* is used to set the fill character back to space. This is necessary because the effects of a call to *fill* last for the duration of the block of code.)

Two other arguments that can be passed to *setf* should also be mentioned at this time. These are *ios::left* and *ios::right*. These affect the justification of an output value within a given width. The default value is *ios::right*.

While an *ios::left* is supposed to be undone by an *ios::right* and an *ios::right* is supposed to be undone by an *ios::left*, this does not happen with some compilers. In order to force an *ios::right* or *ios::left* to be reset to the default condition, a call to *unsetf* can be made. The syntax of *unsetf* is as follows:

```
cout.unsetf(argument);
```

where *argument* is the argument that is to be undone

Example of Left and Right Justification with *unsetf*

```
cout.setf(ios::left);
cout.width(20);
cout << "Chapter 1";
cout.unsetf(ios::left);
cout.width(20);
cout.setf(ios::right);
cout << "The Rise of Rome" << endl;;

cout.setf(ios::left);
cout.width(20);
cout << "Chapter 2";
cout.unsetf(ios::left);
cout.width(20);
cout.setf(ios::right);
cout << "Rome Falls" << endl;
```

// Output for example 1 is as follows:

```
Chapter 1           The Rise of Rome
Chapter 2           Rome Falls
```

Notice that *unsetf* had to be used after sending *ios::left*, but not *ios::right*. It just depends on the compiler you are using if you have to use *unsetf* at all.

By including the library *iomanip*, two other output format functions become available to the programmer. These are *setw* (pronounced "set width") and *setprecision*, which function like *width* and *precision*, except that they are meant to be used as an expression in a `cout` statement. The syntax of these functions are as follows:

setw(#)

where # is the number of spaces the next expression is to be output in

setprecision(#)

where # is the number of decimal places the next expression is to be have on output

Here is an example of the use of *setw* and *precision*.

```
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    cout.setf(ios::fixed, ios::floatfield);
    cout.setf(ios::showpoint);
    double hours=33.5, rate=15.13;
    cout.fill('*');
    cout << "Your pay is $"
         << setw(10) << setprecision(2) << hours * rate << endl;

    return 0;
}
```

// the output for this example is as follows:

```
Your pay is $*****506.86
```

Exercises with Formatting

1) What is the output of the following code?

```
#include <iomanip>
cout.setf(ios::fixed, ios::floatfield);
cout.setf(ios::showpoint);
double d=0.135792468;
cout << setprecision(1) << d << endl;
cout << setprecision(2) << d << endl;
cout << setprecision(3) << d << endl;
cout << setprecision(4) << d << endl;
cout << setprecision(5) << d << endl;
```

2) What is the output of the following code?

```
cout.setf(ios::fixed, ios::floatfield);
cout.setf(ios::showpoint);
double d=0.135792468;
cout.setf(ios::right);
cout.precision(5);
cout << d << endl;
cout.precision(3);
cout << d << endl;
cout.precision(1);
cout << d << endl;
```

3) What is the output of the following code?

```
#include <iomanip>
cout.setf(ios::fixed, ios::floatfield);
cout.setf(ios::showpoint);
double d=0.135792468;
cout.setf(ios::right);
cout.width(10);
cout << setprecision(1) << d << endl;
cout.width(10);
cout << setprecision(2) << d << endl;
cout.width(10);
cout << setprecision(3) << d << endl;
cout.width(10);
cout << setprecision(4) << d << endl;
cout.width(10);
cout << setprecision(5) << d << endl;
```

4) What is the output of the following code?

```
cout.setf(ios::fixed, ios::floatfield);
cout.setf(ios::showpoint);
double d=0.135792468;
cout.setf(ios::right);
cout.precision(5);
cout << setw(10) << d << endl;
cout.precision(3);
cout << setw(10) << d << endl;
cout.precision(1);
cout << setw(10) << d << endl;
```