10 Simple Entry Problems for Data Structures with C++ (Solutions)

CSC 240 Lyon College Fall Term 2024

Marcus Birkenkrahe

October 13, 2024

Entry test: write 10 simple programs (in C++)

1. Declare two variables of type `int`, initialize them to appropriate values, and print them next to one another separated by a comma using the `cout` command and the stream extraction operator `<<`.

   Solution:

   ```
   /* ---------------------------------------- */
   /* Declare, initialize and print two variables */
   /* (CC-BY-NC) Marcus Birkenkrahe 2024          */
   /* ---------------------------------------- */
   // declare and initialize variables x and y
   int x = -10;
   int y = 100;
   // print x, y
   cout << x << ", " << y << endl;
   ```

   ```
   -10, 100
   ```

2. Declare three integer variables: `sum`, `a`, `b`. Initialize `sum` to 0. Initialize the variables `a` and `b` to an appropriate integer value, use an assignment statement to assign `sum` the result of `a` plus `b`, and print the result: `"The sum of _ and _ is _."` using `printf`.

   Solution:

```
/* ---------------------------------------------------------------- */
/* Declare three variables, sum up two of them, and print results  */
/* (CC-BY-NC) Marcus Birkenkrahe modified from Rook's Guide (2013) */
/* ---------------------------------------------------------------- */
// declare three variables sum, a, b and initialize a, b
int sum = 0;
int a = 2, b = 5;
// sum up a and b
sum = a + b;
// print the result
printf("The sum of %d and %d is %d.\n",a,b,sum);


The sum of 2 and 5 is 7.
```

3. Create a program in which 3 variables are declared. Create one float named myFloat, one int named myInt, and one double named myDouble. Initialize them to 3.14, 3, and 3.14159, respectively. Print each variable on a line of its own like this:

```
myFloat  = 3.14
myInt    = 3
myDouble = 3.14159
```

Solution:

```
/* ---------------------------------------------------------------- */
/* Declare 3 variables of different types & print them on one line */
/* (CC-BY-NC) Marcus Birkenkrahe modified from Rook's Guide (2013) */
/* ---------------------------------------------------------------- */
// declare and initialize variables
float myFloat = 3.14;
int myInt = 3;
double myDouble = 3.14159;
// print result
cout << "myFloat  = " << myFloat  << endl
<< "myInt    = " << myInt    << endl
<< "myDouble = " << myDouble << endl;
```

```
myFloat  = 3.14
myInt    = 3
myDouble = 3.14159
```

4. Create a program that displays the diameter and area of a circle for any given radius. Initialize the radius in the program. Use a `const float` to represent the literal . The output should look like this: `The area of a circle of diameter 2 is 3.14159` (for r=1).

   Solution:

```
/* ---------------------------------------------------------------- */
/* With constant Pi, compute area of circle for given radius       */
/* Sample input: r = 1.                                             */
/* Sample output: The area of a circle of diameter 2 is 3.14159    */
/* (CC-BY-NC) Marcus Birkenkrahe modified from Rook's Guide (2013) */
/* ---------------------------------------------------------------- */
// const declaration
const float pi = 3.141592;
// variable declarations
float area;
// initialize variables
float radius = 1.f; // radius = 1
// compute area
area = pi * radius * radius;
// print results
cout << "The area of a circle of diameter " << 2 * radius << " is " << area << enc


The area of a circle of diameter 2 is 3.14159
```

5. Comment each line of this code:

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

int main()
{
  int time;
  cout << "Enter time in seconds:\n";
```

```
  cin >> time;
  cout << "You entered: " << time << " seconds." << endl;
  int answer = (32 * time * time) / 2;
  cout << "The distance is ";
  cout << answer;
  cout << " feet.\n";
  return 0;
}
```

Background: In the imperial metric system, 32 feet per second squared (or 9.8 meter per second squared in the metric system) is the approximate acceleration due to Earth's gravity for a freely falling body (no air friction or other effects assumed).

Solution:

```
/* ------------------------------------------------------------- */
/* Compute distance of a freely falling body with constant       */
/* acceleration a = 32 ft/s^2 due to gravity, d = a t^2 / 2      */
/* Sample input: time = 10.                                      */
/* Sample output: You entered: 10 seconds. Distance is 1600 feet */
/* (CC-BY-NC) Marcus Birkenkrahe modified from Rook's Guide (2013) */
/* ------------------------------------------------------------- */
// Load input output header file
#include <iostream>
// use standard namespace
using namespace std;
// main function - no arguments, returns integer
int main()
{ // body of main function begins
  // declare integer variable `time`
  int time;
  // Ask for user input
  cout << "Enter time in seconds:\n";
  // Stream user input from standard device to variable `time`
  cin >> time;
  // Tell user which number he entered.
  cout << "You entered: " << time << " seconds." << endl;
  // declare and initialize variable `answer` with arithmetic expression
  int answer = (32 * time * time) / 2;
  // print result `answer` over one line ending with a new line
```

```
  cout << "The distance is ";
  cout << answer;
  cout << " feet.\n";
  // return 0 from main function
  return 0;

} // body of main functions ends

echo "10" > data/input
cat data/input
```

6. Data Types and conversion

   Write a 3-line program that declares a variable named `sampleSize` and set it to 14.58093. Use a compound operator to increase its value by 12.495. Finally print the result converted to an integer using `cout` and `int`. The output should be 27.

   Solution:

```
/* ---------------------------------------------------------------- */
/* Declare, initialize, change, & print a floating-point variable  */
/* converted to an integer using a compound operator, cout and int */
/* Const input: sampleSize = 14.58093, adding 12.495               */
/* Sample output: Integer sampleSize: 27                           */
/* (CC-BY-NC) Marcus Birkenkrahe modified from Rook's Guide (2013) */
/* ---------------------------------------------------------------- */
// declare variable
double sampleSize = 14.58093;
// increase value
sampleSize += 12.495;
// print result converted to integer
cout << "Integer sampleSize: " << int(sampleSize) << endl;


Integer sampleSize: 27
```

7. Conditionals

   - Write a program that declares two integers a and b and initializes them with appropriate values. The program should print out one

message that informs the user if a is smaller or bigger than b, or if they're the same.

Solution:

```
/* ------------------------------------------------------------- */
/* Declare two integers a and b, commpare them and print out if  */
/* they are the same or if a is bigger or smaller than b.        */
/* Sample output: a and b are the same!                          */
/* (CC-BY-NC) Marcus Birkenkrahe modified from Rook's Guide (2013) */
/* ------------------------------------------------------------- */
// Declare variables
int a = 100, b = 10;
// Compare values
if (a == b) {
  cout << a << " and " << b << " are the same!" << endl;
 } else if (a > b) {
  cout << a << " is bigger than " << b << "!" << endl;
 } else {
  cout << a << " is smaller than " << b << "!" << endl;
 }
```

- Rook's Guide to C++ contains this rather useless (though not wrong) flow chart (fig. 10.2, pg. 51). This chart can be improved a lot, see this BPMN model created at bpmn.io (link).

8. Loops

   Do the first program, if you've completed section 7, and do the second program, if not.

   (a) Wrap the program 'compare and b' from sect. 7 ("Conditionals") in an infinite loop, and ask the user after each iteration if he wants to quit or continue playing, exit the program accordinly, and print the number of iterations. Play at least once.

   Solution:

```
/* ------------------------------------------------------------- */
/* In an infinite loop: Ask user to enter two integer numbers, then  /*
 /* compare them. At the end of each iteration, ask if user wants to  /*
 /* quit. (CC-BY-NC) Marcus Birkenkrahe modified from Jensen (2013)   /*
 /* ------------------------------------------------------------- */
// Declare variables
```

```
int a, b;
char quit;
// infinite loop
do {
// Ask for user input
cout << "Enter two numbers: ";
// Store input in variables
cin >> a >> b;
// check for valid input
if (cin.fail()) {
  cout << "Invalid input. Please enter two integers!" << endl;
  break;
}
// Compare values
if (a == b) {
  cout << a << " and " << b << " are the same!" << endl;
} else if (a > b) {
  cout << a << " is bigger than " << b << "!" << endl;
} else {
  cout << a << " is smaller than " << b << "!" << endl;
}
cout << "Quit playing? Enter Y: \n";
cin >> quit;
 } while (quit != 'Y');
cout << "Done" << endl;
```

Testing with sample data:

```
echo "100 100 N
   -100 100 N
      8   1 Y" > data/compare
cat data/compare
```

(b) Create a `for` loop that outputs your `name` to the screen 10 times before exiting the loop.

Solution I:

```
for (int i=0; i<10; i++) {
cout << "Marcus" << endl;
 }

Marcus
```

```
Marcus
Marcus
Marcus
Marcus
Marcus
Marcus
Marcus
Marcus
Marcus
```

Solution II (storing the name as a `string` type):

```
#include <string> // include string library

string name = "Marcus"; // set name string variable

for (int i=0; i<10; i++) {
cout << name << endl;
 }
```

```
Marcus
Marcus
Marcus
Marcus
Marcus
Marcus
Marcus
Marcus
Marcus
Marcus
```

9. Arrays

   Create a program in which an integer array named `myArray` is declared with a size of 10. Use a `for` loop to prompt the user to store a value in every index of the array. Aer the array is given values, output the values of the array to the screen using a `for` loop. Output each value of the array on its own line.

   Input: 10 integers

```
echo "4 56 7 324 -4 0 21 -999 9 1" > data/array
```

```
cat data/array
cat data/array | wc -w
```

Solution:

```
/* ------------------------------------------------------------- */
/* Declare an integer array of size 10 & prompt user to store a  */
/* value in every index of the array using a for loop            */
/* Output: array elements one per line.                          */
/* (CC-BY-NC) Marcus Birkenkrahe modified from Rook's Guide (2013) */
/* ------------------------------------------------------------- */
int myArray[10]; // declare integer array of length 10

// initialize array values
for (int i = 0; i < 10; i++) {
  cin >> myArray[i];
 }

// Output array elements one per line
for (int i : myArray) cout << i << endl;
```

10. Functions

    Write code that prompts the user for a number of miles travelled and a number of hours, then calculates the user's speed in miles per hour using a user-defined function named `mph`.

    If you're doing this in Emacs, use the complete C++ program header and call `mph` in a `main` function:

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

int main() {
    //....
}
```

    Solution:

```
/* ------------------------------------------------------------- */
/* Compute speed based on miles travelled and number of hours    */
```

```
/* User input: miles, hours                                          */
/* Output: With __ miles in __ hours, your average speed was __ mph */
/* (CC-BY-NC) Marcus Birkenkrahe modified from Rook's Guide (2013)  */
/* ---------------------------------------------------------------- */
#include <iostream>

double mph(double miles,double hours) {
  return miles / hours;
}

int main() {

  // variable declarations
  double milesTravelled, hoursTravelled;

  // Get user input
  cout << "Enter miles and hours travelled: ";
  cin >> milesTravelled >> hoursTravelled;
  cout << endl;

  //compute and print result
  cout << "With " << milesTravelled << " miles in "
   << hoursTravelled << " hours, your speed was "
   << mph(milesTravelled,hoursTravelled) << " mph." << endl;
  return 0;
}
```

Testing:

```
echo "740 11.5" > data/mph
cat data/mph
```