Description

The purpose of this lab is to check your understanding of the content in lecture 4a, Arrays and Pointers.

We will focus on the material from chapter 7 of the book. The objectives are:

- To use the array data structure to represent a set of related data items.
- To use arrays to store, sort and search lists and tables of values.
- To declare arrays, initialize arrays and refer to the individual elements of arrays.
- To pass arrays to functions.
- Basic searching and sorting techniques.
- To declare and manipulate multidimensional arrays.
- To use C++ Standard Library class template vector.

Part 1: Questions

Complete the assignment in the module *Topic 4 Lab: Arrays and Pointers* named, "Lab 4a Questions", in Canvas.

Part 2: Activity

This part of the lab should be submitted using the Lab 4a Activity assignment in the *Topic 4a Lab: Arrays and Pointers* module.

You should complete the non-programming parts of this lab using this document. It is possible that some of the images and answer-boxes below might move or need to be resized while using them. Do your best to make your final document neat and organized.

Problem 1: Program Output

For each of the given program segments, read the code and write the output in the space provided below each program. [Note: Do not execute these programs on a computer.]

1. What is output by the following program segment?

```
1  int i;
2  int values[ 10 ] = { 4, 1, 1, 3, 4, 9, 9, 2, 1, 7 };
3
4  cout << "Element" << setw( 13 ) << "Value" << endl;
5
6  for ( i = 0; i < 10; i++ )
7     cout << setw( 7 ) << i << setw( 13 ) << values[ i ] << endl;
```

```
Answer
Element             Value
      0                 4
      1                 1
      2                 1
      3                 3
      4                 4
      5                 9
      6                 9
      7                 2
      8                 1
      9                 7
```

2. What is output by the following code segment?

```
1  char string1[] = "How are you?";
2
3  cout << "string1 is: " << string1 << endl;
4
5  for ( int i = 0; string1[ i ] != '\0'; i++ )
6     cout << string1[ i ] << "_";
```

```
Answer
string1 is: How are you?
H_o_w_ _a_r_e_ _y_o_u_?_
```

3. What is output by the following program?

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

void mystery();

int main()
{
   cout << "First call to mystery:\n";
   mystery();

   cout << "\n\nSecond call to mystery:\n";
   mystery();
   cout << endl;
} // end main

// function mystery definition
void mystery()
{
   static int array1[ 3 ];
   int i;

   cout << "\nValues on entering mystery:\n";

   for ( i = 0; i < 3; i++ )
      cout << "array1[" << i << "] = " << array1[ i ] << "  ";

   cout << "\nValues on exiting mystery:\n";

   for ( i = 0; i < 3; i++ )
      cout << "array1[" << i << "] = "
           << ( array1[ i ] += 2 ) << "  ";

} // end function mystery
```

```
Answer
First call to mystery:

Values on entering mystery:
array1[0]=0 array1[1]=0 array1[2]=0
Values on exiting mystery:
array1[0]=2 array1[1]=2 array[2]=2

Second call to mystery:

Values on entering mystery:
array1[0]=2 array1[1]=2 array1[2]=2
Values on exiting mystery:
array1[0]=4 array1[1]=4 array1[2]=4
```

4. What is output by the following program?

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

int main()
{
   int array[ 3 ][ 4 ] = { { 1, 2, 3, 4 }, { 2, 3, 4, 5 }, { 3, 4, 5, 6 } };

   for ( int i = 0; i < 3; i++ )
   {
      for ( int j = 0; j < 4; j++ )
      {
         cout << array[ i ][ j ] << " ";
      }

      cout << endl;
   }
} // end main
```

```
Answer
1 2 3 4
2 3 4 5
3 4 5 6
```

Problem 2: Correct the Code

For each of the given program segments, determine if there is an error in the code. If there is an error, specify whether it is a logic or compilation error, circle the error in the program, and write the corrected code in the space provided after each problem. If the code does not contain an error, write "no error." [*Note:* It is possible that a program segment may contain multiple errors.]

1. The following code should assign 8 to the fifth element in *array*:

```cpp
array[ 5 ] = [ 8 ];
```

```
Answer compilation error
array[5] = 8;
```

2. The for loop should initialize all array values to -1.

```cpp
int array[ 10 ];

for ( int i = 0; i < 9; i++ )
   array[ i ] = -1;
```

```
Answer logic error
int array[10]

for(int i = 0; i <= 9; i++)
     array[i] = -1;
```

3. Array *array* should contain all the integers from 0 through 10, inclusive.

```
1  int array[ 10 ] = { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
```

```
Answer logic error & compilation
int array[11] = {0,1,2,3,4,5,6,7,8,9,10};
```

4. The following code segment should declare two arrays containing five and six elements, respectively:

```
1  const int arraySize = 5;
2  int a[ arraySize ];
3
4  arraySize = 6;
5
6  int b[ arraySize ];
```

```
Answer compilation error
array<int, 5> a;

array<int, 6> b;
```

5. The for loop that follows should print *array*'s values:

```
1  int array[ 10 ] = { 0 };
2
3  for ( int i = 0; i <= 10; i++ )
4     cout << array[ i ];
```

```
Answer logic error
int array[10] = {0};

for(int i =0; i <= 9; i++)
      cout << array[i];
```

6. The for loop that follows should print all of array's values:

```
1  int array[ 3 ][ 3 ] = { 1, 2, 3, 4, 5, 6, 7, 8, 9 };
2
3  for ( int i = 0; i < 3; i++ )
4
5     for ( int i = 0; i < 3; i++ )
6        cout << array[ i ][ i ];
```

```
Answer compilation and logic error
int array[3][3] = {1,2,3,4,5,6,7,8,9};

for(int i = 0; i<3; i++)

        for(int j=0; j<3; j++)
                cout << array[i][j];
```

7. Research and answer the question, what are the benefits of using the *std::array*? What are the downsides?

Answer
Some of the benefits of using std::array over the built in array are that they will maintain the size of the array because it is fixed. It also prevents automatic decay into a pointer and provides bounds checking unlike the built in array. The downsides are that they cannot be expanded or contracted dynamically.

Problem 3: Programming

1. Write a program that simulates the rolling of two dice.
    - **Call the program, "dice.cpp".**
    - The program should call rand to roll the first die, and should call rand again to roll the second die.
    - The sum of the two values should then be calculated. [Note: Each die has an integer value from 1 to 6, so the sum of the two values will vary from 2 to 12, with 7 being the most frequent sum and 2 and 12 being the least frequent sums.]
    - There 36 possible combinations of the two dice.
    - Your program should roll the two dice 360,000 times.
    - Use a one-dimensional **std::array** to tally the numbers of times each sum appears.
    - Print the results in a tabular format. **Follow the sample output exactly.**
    - Also, determine if the totals are reasonable (i.e., there are six ways to roll a 7, so approximately one sixth of all the rolls should be 7).
    - Use the *at* member function of the array class to access the elements of the array.
    - **Be sure to use constant variables for the dimension of the array.**

Sample Output:

```
Sum     Total  Expected   Actual
  2      1000    2.778%    2.778%
  3      1958    5.556%    5.439%
  4      3048    8.333%    8.467%
  5      3979   11.111%   11.053%
  6      5007   13.889%   13.908%
  7      6087   16.667%   16.908%
  8      4996   13.889%   13.878%
  9      3971   11.111%   11.031%
 10      2996    8.333%    8.322%
 11      2008    5.556%    5.578%
 12       950    2.778%    2.639%
```

2. Modify the program to use a two-dimensional **std::array** and call it, **"dice_2d.cpp"**.
    - Rather than counting the number of times each sum appears, increment the correct cell in the array that represents the first die's value (the row) and the second die's value (the column).
    - **Be sure to use constant variable(s) for the dimensions of the array.**
    - Print this array with the number of times each dice combination occurred. **The output must look like the following**:

	1	2	3	4	5	6
1	1011	971	1027	1025	971	1015
2	1013	968	990	968	1081	993
3	993	1014	983	973	1019	977
4	980	1004	974	1022	946	1046
5	1003	1021	1019	979	1004	1056
6	1026	1015	931	989	1014	979

What to Submit for Lab 4a Activity

- A PDF of this document completed.
- dice.cpp
- dice_2d.cpp
- A screenshot of the **output for *each program***.
  Note: your screenshots must include the entire Visual Studio Code window. Do not include your desktop or anything else in the image. Do not take a picture with a camera or phone, use your computer to create a screenshot.