

## Hands-on Activity 4.2

### Arrays

Course Code: CPE007

Program: Computer Engineering

Course Title: Programming Logic and Design

Date Performed: 9/11/2025

Section: CPE11S1

Date Submitted: 9/13/2025

Name(s): Juan Paulo C. Lara

Instructor: Engr. Jimlord M. Quejado

### 6. Output

Example 1

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5      int n[10];
6
7      // Initialize array elements to 0
8      for (int i = 0; i < 10; i++) {
9          n[i] = 0;
10     }
11
12     cout << "Element   Value" << endl;
13
14     // Print index and value
15     for (int i = 0; i < 10; i++) {
16         cout << "      " << i << "      " << n[i] << endl;
17     }
18
19     return 0;
20 }
21
```

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

-----  
Process exited after 0.01101 seconds with return value 0  
Press any key to continue . . . |

During execution of this sample program, it starts by line 5 declaring an array of 10 integers to variable n. The first for loop at line 8 scans the i array from 0 to 9 and assigns them at 0. This makes the output of values all zero. Line 12 prints out a column header to label the following numbers or data in the next processes. The next for loop scans from 0 to 9 to print the index numbers from the 10 values in the array. Next line, It then prints the values placed in n[i] starting from 0 to 9 in a column along with i. The blank spaces in quotations are placed to line up with the column header.

#### Example 2

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5      int n[10] = {32, 27, 64, 18, 95, 14, 90, 70, 60, 37};
6
7      cout << "Element  Value" << endl;
8
9      for (int i = 0; i < 10; i++) {
10         cout << "    " << i << "    " << n[i] << endl;
11     }
12
13     return 0;
14 }
```

Element	Value
0	32
1	27
2	64
3	18
4	95
5	14
6	90
7	70
8	60
9	37

-----  
Process exited after 0.01478 seconds with return value 0  
Press any key to continue . . . |

During execution of this sample code, line 5 first initializes and declares a 10 number array to n with the values 32, 27, 64, 18, 95, 14, 90, 70, 60, 37 from index 0 to 9. The following line prints out a column header to label the numbers after processing. A for loop at line 9 scans i from index 0 to 9 and counts the times the values are assigned. During the loop, variable i shows the index in the array while n[i] assigns the given values. Then, these values are printed 10 times with each of the assigned values during the process. Blanks spaces and endl are used to form a column of values.

#### Example 3

```

1  #include <iostream>
2  using namespace std;
3
4  #define SIZE 12
5
6  int main() {
7      int a[SIZE] = {1, 3, 5, 4, 7, 2, 99, 16, 45, 67, 89, 45};
8      int total = 0;
9
10     for (int i = 0; i < SIZE; i++) {
11         total += a[i];
12     }
13
14     cout << "Total of array element values is " << total << endl;
15     return 0;
16 }

```

```
Total of array element values is 383
```

```
-----
Process exited after 0.01185 seconds with return value 0
Press any key to continue . . . |
```

During execution, line 4 defines a value of 12 to variable SIZE to work as a constant value. Line 7 initializes a with 12 elements and declares the array with the following numbers: 1, 3, 5, 4, 7, 2, 99, 16, 45, 67, 89, 45. Line 8 stores the variable total as a sum of all given elements inside the array. At line 10, one for loop scans i counting from 0 to 11. While looping, a[i] looks for values stored inside the array and adds them to variable total. And, after adding up the values during the loop, line 14 outputs the result of the addition.

## 7. Supplementary Activity

Problem 1

```

1  #include <iostream>
2  using namespace std;
3  int main(){
4      int n[10]={19, 3, 15, 7, 11, 9, 13, 5, 17, 1};
5
6      cout << "Element Value   Histogram" << endl;
7
8      for (int i = 0; i < 10; i++){
9          cout << "      " << i << "      \t " << n[i] << "\t";
10
11         for (int j = 0; j < n[i]; j++){
12             cout << "*";
13         }
14         cout << endl;
15     }
16     return 0;
17 }

```

```

Element Value   Histogram
0      19      *****
1       3       ***
2      15      *****
3       7       *****
4      11      *****
5       9       *****
6      13      *****
7       5       *****
8      17      *****
9       1       *

```

```

-----
Process exited after 0.0179 seconds with return value 0
Press any key to continue . . . |

```

Compared to the previous programs, this program has an additional feature, a histogram to represent the scale of the numbers. During execution, line 4 initializes n with an array containing the 10 elements with the following values 19, 3, 15, 7, 11, 9, 13, 5, 17, 1. The next line prints a column header with the spacings to line up with the numbers. The outer for loop at line 8 scans the indexes from 0 to 9 to print the given values of each element row by row. The inner loop at line 11 loops for the amount of times given by the values at the array to print the \* symbols at a certain amount. After that, line 14 at the outer loop acts as a stop and go to get the next element. The labels of the printed header Element represents the index in i, Value represents a number stored in n[i], Histogram represents the repeated \* symbols based on the values inside the array.

```

1  #include <iostream>
2  using namespace std;
3
4  #define RESPONSE_SIZE 40
5  #define RESPONSE_LIMIT 10
6
7  int main()
8  {
9      int responses[RESPONSE_SIZE] = { 1, 2, 6, 4, 8, 5, 9, 7, 8, 10, 1, 6, 3, 8,
10     6, 10, 3, 8, 2, 7, 6, 5, 7, 6, 8, 6, 7, 5, 6, 6, 5, 6, 7, 5, 6, 4, 8, 6, 8, 10
11     };
12
13     int frequency[RESPONSE_LIMIT + 1] = {0};
14
15     for (int i = 0; i < RESPONSE_SIZE; i++)
16     {
17         int response = responses[i];
18         frequency[response]++;
19     }
20
21     cout << "Response Summary: " << endl;
22     for (int i = 1; i <= RESPONSE_LIMIT; i++)
23     {
24         cout << "Response " << i << ": " << frequency[i] << " students" << endl;
25     }
26
27     return 0;
28 }

```

```

Response Summary:
Response 1: 2 students
Response 2: 2 students
Response 3: 2 students
Response 4: 2 students
Response 5: 5 students
Response 6: 11 students
Response 7: 5 students
Response 8: 7 students
Response 9: 1 students
Response 10: 3 students

```

```

-----
Process exited after 0.02061 seconds with return value 0
Press any key to continue . . . |

```

During execution of this code, first, line 4 and 5 constantly defines RESPONSE\_SIZE AS 40 and RESPONSE\_LIMIT AS 10. Line 9 declares the values of 40 students to an array. The array at line 13 acts as a counter for the frequency of responses of the numbers 1-10. It is then added by + 1 to ignore the index 0. The first for loop scans the index from 0 to 39, then it adds 1 to each response. After the iteration, the array named frequency holds the numbers chosen by the students. At line 21, a column header is printed. Next line, another for loop prints the chosen response and the number of students.

## 8. Conclusion

Concluding this activity, I learned more about the for loops and how they're used to store, display, and manipulate values in line. More specifically the #define command which is similar to the standard method of declaring values with int but it's

something that acts more like a constant to keep the values the same for each loop. In the histogram, I used another for loop that counts for the values in the array to represent the scale of the given values by repeating the \* symbol. With this new technique, I can change the given values without going through each variable and changing them, creating a simpler program but with more features than the primitive yet larger programs (in terms of the amount of lines) I wrote earlier.

## **9. Assessment Rubric**