

## Seatwork 4.2

### Arrays

Course Code: CPE007

Program: Computer Engineering

Course Title: Programming Logic and Design

Date Performed: September 11, 2025

Section: CPE11S1

Date Submitted: September 11, 2025

Name(s): Lopez, Andrei Dion C.

Instructor: Engr. Jimlord 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 }
```

In this code the first for loop makes it so that all of the values in each element is the value of 0, this also makes it so that the code won't output an error stating that the values in each element hasn't been initialized. Then it prints out the identifiers of the values and then does a second loop to print out the index number and the value that is stored in that index value in a vertical line format to make it easy to read.

Output:

```
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 0.04921 seconds with return value 0
Press any key to continue . . . |
```

## 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 }
```

Similarly to the code in the example above this prints out the index and the values written in a vertical format, what makes this code different is the initialization of the values in at the start of the code at line 5 instead of making a loop that would set a value of 0 to each of the elements in the array.

Output:

```
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 0.01468 seconds with return value 0
Press any key to continue . . . |
```

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 }
```

What this example shows is the addition of all elements inside an array. The way it does this is by using a *for* loop. In the *for* loop, it initializes an integer that is treated as the *n*th number of the array, and what it does is that it takes the number from the array, and adds it to the total. The integer is then increased by an increment until it meets the size value, looping the entire block of code until all the numbers in the array are added up to the total variable.

Output:

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

```
-----
Process exited after 0.00799 seconds with return value 0
Press any key to continue . . . |
```

## 7. Supplementary Activity

### Activity No. 1

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5      int num[10] = {19, 3, 15, 7, 11, 9, 13, 5, 17, 1};
6      char j = '*';
7
8
9      cout << "Element  Value  Histogram" << endl;
10
11     for (int i = 0; i < 10; i++) {
12         cout << "    " << i << "    " << num[i] << "    ";
13         for (int x = 0; x < num[i]; x++) {
14             cout << j;
15         }
16         cout << endl;
17     }
18
19     return 0;
20 }
```

This code shows the amount of “\*” an element can show by running a second for loop inside the first for loop allows the computer to repeatedly output the character until x reaches the same number that the element is. Once it does this it will move on to the next element value and repeat.

Output:

```
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 0.02232 seconds with return value 0
Press any key to continue . . . |
```

## Activity No. 2

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5      const int res_size = 40;
6      const int Fsize = 11;
7
8      int responses[res_size] = {1, 2, 6, 4, 8, 5, 9, 7, 8, 10,
9                                1, 6, 3, 8, 6, 10, 3, 8, 2, 7,
10                               6, 5, 7, 6, 8, 6, 7, 5, 6, 6,
11                               5, 6, 7, 5, 6, 4, 8, 6, 8, 10};
12
13      int freq[Fsize] = {0};
14
15
16      for (int i = 0; i < res_size; i++) {
17          ++freq[responses[i]];
18      }
19
20      cout << "Response Summary:\n";
21      cout << endl;
22
23      for (int i = 1; i < Fsize; i++) {
24          cout << "Response " << i << ": " << freq[i] << " students" << endl;
25          cout << endl;
26      }
27
28      return 0;
29 }
```

At lines 5 and 6 I initialized the integers as constants so that their values are unchangeable, next I initialized a second array to store the values of the elements. For the *for* loop, the variable that is initialized counts up until it reaches the amount of responses in the array, and every time it loops, it adds an increment to the frequency variable. The next *for* loop then shows the amount of students that responded to the exact value.

Output:

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 0.1755 seconds with return value 0  
Press any key to continue . . .

## 8. Conclusion

In this Seatwork I learnt how to print out arrays in different possible ways like its position in the array. I also learnt how to summarize the amount of similar responses in an array.

## 9. Assessment Rubric