Activity #16: Arrays

Recorder's Report

Manager: Reader:

Recorder: Driver:

Date: Score: Satisfactory / Not Satisfactory

Record your team's answers to the key questions (marked with ) below.

a) Model 1, Question #6

b) Model 2, Question #12.c

c) Model 3, Question #20

Activity #16: Arrays

In this course, you will work in teams of 3–4 students to learn new concepts. This activity will introduce you to arrays in C++.

Content Learning Objectives

After completing this activity, students should be able to:

- Explain the difference between vector and array syntax in C++
- Identify one- and two-dimensional arrays and understand their syntax in C++
- Understand how one- and two-dimensional arrays are passed to functions in C++

Process Skill Goals

During the activity, students should make progress toward:

- Translate code between vectors and one-dimensional arrays in C++
- Write code to manipulate two-dimensional arrays in C++

Model 1 One Dimensional Arrays

```
string homeworkNames[5] = {"Variables","If-Else","Loops","Vectors","Arrays"};
int pointsPossible[5] = {25, 35, 15, 20, 30};
int numStudents = 40;

for (int i=0; i<5; i++) {
  cout << homeworkNames[i] << ": " << pointsPossible[i] << endl;
}
cout << pointsPossible[5] << endl;

```

Refer to Model 1 above as your team develops consensus answers to the questions below.

Questions (15 min) Start time:

1. Like vectors, arrays can be used to store a list of elements of the same type. Answer the following questions about the arrays in the model above.

a) What are the names of the two arrays used in the model?

b) What are the types of the two arrays in the model?

c) What are the sizes of the two arrays in the model?

d) What word describes the relationship between the arrays?

e) How would you access the last element of `homeworkNames`?

2. Based on the model, describe how the following symbols are used with arrays.

a) The curly braces, `{ }`.

b) The square brackets, `[ ]`. Give two different ways they are used.

3. Without running it, predict what the code from this model will print out.

4. This code can be found in `activity16a.cpp`. Run it and explain what you see.

5. What happens if you change line 8 to `cout << pointsPossible`?

6. Suppose that you were asked to rewrite this program to use vectors instead of arrays. 

   a) How would you change lines 1 and 2 to declare vectors instead of arrays?

   b) How would you change line 6 to print out the vector values?

   c) Suppose line 8 was changed to `cout << pointsPossible.at(5) << endl;`. What would happen when the program runs that command?

7. The code below is designed to collect 100 names and then print out a random one. It is built using vectors. Rewrite it using arrays.

```
vector<string> names;
string tmpName;
for (int i = 0; i < 100; i++) {
  cout << "Name " << (i + 1) << ": ";
  cin >> tmpName;
  names.push_back(tmpNames);
}
cout << names.at(rand() % 100) << endl;

```

Model 2 Two Dimensional Arrays

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

Output:

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

Refer to Model 2 above as your team develops consensus answers to the questions below.

Questions (20 min) Start time:

8. A *two dimensional array* (2D array) uses two indices to identify entries. What is the difference in C++ syntax between declaring a 2D array and a one-dimensional (1D) array?

9. By convention, the first index is used for the row number and the second index is used for the column number. Find the value of each entry in the 2D array given in the model.

a) What is the value of `table[2][1]` in the code above?

b) What is the value of `table[1][0]` in the code above?

c) What is the value of `table[0][1]` in the code above?

10. Suppose a new 2D array was declared as `char tableTwo[3][3]`. Use square brackets to fill in the indices for each of the elements. The first one is done for you.

`tableTwo[0][0] = 'A'`	`= 'B'`	`= 'C'`
`= 'D'`	`= 'E'`	`= 'F'`
`= 'G'`	`= 'H'`	`= 'I'`

11. Consider how you would write code to print the first row of `tableTwo` above. That is, to print `A B C`.

a) First give two lines of similar code that will print out the first two elements, `A B`.

b) Suppose the first row had 200 elements. Give a few lines of code to print the entire first row.

c) Suppose there were `int col` elements per row. Give code to print out the `int k`th row.

12. Now consider how you would adjust your code above to print out columns instead of rows.

a) First give two lines of code that will print out the first two elements of the first column, `A D`.

b) Suppose the first column had 200 elements. Give code to print the entire first column.

c) If there were `int row` elements per column, write code to print out the `int k`th column.

13. The file `activity16b.cpp` contains code that initializes a 2D array with 10 rows and 10 columns, whose entries are integers from 1 to 100. Use a nested for loop to print out the following entries in the array. You may wish to use `setw(4)` from the `iomanip` library.

```
 1
11  12
21  22  23
31  32  33  34
41  42  43  44  45
51  52  53  54  55  56
61  62  63  64  65  66  67
71  72  73  74  75  76  77  78
81  82  83  84  85  86  87  88  89
91  92  93  94  95  96  97  98  99 100

```

Model 3 Arrays as Function Parameters

```
1  const int NUM_ROWS = 3;
2  const int NUM_COLS = 3;
3
4  // initialize the game board to spaces
5  void initBoard(char board[][NUM_COLS], int rows);
6
7  // print the game board
8  void printBoard(char board[][NUM_COLS], int rows);
9
10 // prompt player `turn' for row/col of play
11 void getMove(char turn, int &row, int &col);
12
13 // check if single array isall the same char
14 bool isWinningRow(char row[], int size, char &winner);
15
16 // check if a board has a winning row/col/diagonal
17 bool isWinningBoard(char board[][NUM_COLS],
18                     int rows, char &winner);
19
20
```

```
1  int main() {
2    char board[NUM_ROWS][NUM_COLS];
3    char winner,playAgain;
4    int row, col;
5    do {
6      char turn = 'O';
7      initBoard(board,NUM_ROWS);
8      do {
9        if(turn == 'X') turn = 'O'; else turn = 'X';
10       printBoard(board,NUM_ROWS);
11       do {
12         getMove(turn,row,col);
13       } while( board[row][col] != ' ');
14       board[row][col] = turn;
15     } while(!isWinningBoard(board,NUM_ROWS,winner));
16     printBoard(board,NUM_ROWS);
17     if (winner != 'D') {
18       cout << "Congrats player " << winner << endl;
19     } else {
20       cout << "It's a draw!" << endl;
21     }
22     cout << "Play again (y/n)? ";
23     cin >> playAgain;
24   } while (playAgain == 'y');
25   return 0;
26   }
27
```

Refer to Model 3 above as your team develops consensus answers to the questions below.

Questions (15 min) Start time:

14. Determine if the functions prototyped in the model above have 1D, 2D, or no array parameters.

a) `void initBoard(...)`

b) `void printBoard(...)`

c) `void getMove(...)`

d) `bool isWinningRow(...)`

e) `bool isWinningBoard(...)`

15. Based on your observations above, answer the following questions about passing arrays to functions.

a) How do you specify that a given parameter is a 1D array?

b) How do you specify that a given parameter is a 2D array?

c) How do you pass an array as an argument in a function call?

d) What other parameter must you always pass with an array parameter?

16. This program, together with the function definitions, is in `activity16c.cpp`. What does it do?

17. How is the 2D array defined on line 2R used in this program? In particular, suppose the following output was produced by `printBoard()`. What would the values of each array entry be?

```
1   O | O | X
2  ---+---+---
3     | X |
4  ---+---+---
5     |   |
6
```

18. What happens if you change the values of the constants on lines 1L and 2L from `3` to `4`?

19. Why do you think we always have a size parameter together with any array parameter of a function?

20. Take another look at the function `void initBoard` declared in this model.

a) What does this function do?

b) On what line is the `char board[][]` variable passed to this function declared?

c) Is the `char board[][]` variable modified in this function local or global?

d) Based on the effect of the function, was the array passed by value or by reference?

e) Was an `&` used in the function header to make the array pass-by-reference?

f) What can you conclude about how array parameters are passed?