

CSE1301 Exercise Sheet 6

Arrays

Exercise 1

Write C functions that perform the following tasks:

- a) Read in a sequence of integers and store them in an array. Incorporate test statements to make sure the input is valid.
- b) Print the array elements separated by blanks.
- c) Add the elements of the array and print the sum.

Write a “driver” program that will call these functions.

Plan test data for your program.

Exercise 2

Write a function that takes an array of doubles, an integer representing the length of the array, and a double as parameters and multiplies every element of the array by the double.

Write a program to test your function with various arrays of different lengths.

Exercise 3

Suppose that you are going to play 18 holes of golf with two friends. Write a C program to input the score for each hole (for you and both your friends), and then at the end of the round (i.e. after hole number 18), add up the scores and report on various aspects for each of you.

Your program should use arrays to store 18 scores for an 18 hole round. One 1-D array should store the so-called ‘par’ score for each hole, where par is the number of shots a player is expected to take for a given hole. You can choose for yourself what the par score is for each of the 18 holes (par is usually 3 or 4 or 5 shots). You should also use a 3-by-18 2D array, to keep your own score and the scores for each of your two friends.

For each player, the program should report on the following: the total score for the round, the average number of shots per hole, and how many of each of the following were achieved:

- Hole in 1
- Par
- Birdie (one under par)
- Eagle (two under par)
- Albatross (three under par)
- Bogey (one over par)
- Double bogey (two over par)
- Triple bogey (three over par)

Your program should also report the average score for each of the 18 holes. For example, for hole number 8, if you took 4 shots, one friend 6 shots and another 8 shots, the average for that hole would be 6.

Before you start coding, do a top-down design for the program. You should use at least two functions (one for inputting the scores, and one for printing the post-round report).

You should also take into consideration that your program should be easy to test. In particular, you might find it a hassle that there are so many holes! Use a #define to specify the number of holes and test your program initially with a three hole course. Only run tests over all 18 holes when you are convinced that your program works independently of the number of holes defined.