5 Athletes

Red Opal Innovations

Version 1

Table of Contents

[Purpose 3](#_Toc83553115)

[Background 3](#_Toc83553116)

[Testing Data 4](#_Toc83553117)

[Content control 6](#_Toc83553118)

[update Table of Contents when finished]

# 

# Purpose

The purpose of this document is to produce a test plan for the 5 athletes’ scripts.

## Background

The 3 JavaScript scripts will allow you to enter 5 Athletes names and heights into an array.

They also manipulate the array.

Task1.js and task2.js are both console applications. They both declare the array and asks the user for 5 names and heights, each in a set. One athlete at a time, the name and the height are entered. Once the array has the 5 names and heights a message is displayed that the array has been filled. A prompt is displayed asking for a single number input of the array item number to display the athlete and height in that array position.

The last part of the script is the deletion of a name and a height from the array. A prompt is shown and asks for an athlete’s name and the once the script locates the name and the height for the name and removes them both from the array. The difference between task1.js and task2.js is the function called to delete the athlete’s name.

Task3.js is used in conjunction with taskthree.html. We are presented with the following form:  
  
Rectangle

Description automatically generated with low confidence

We enter the name and the height of the athlete to be stored in the array. We can add up to 5 athletes. Once we enter the name, we submit to the array using the save button. A message is displayed saying if the name and height has been saved or if the array is full.

The find button will search the array and give us the position within the array of the athlete found. If nothing is found, we are given an error.

# Testing Data

Here is a table of the data set we will be using for the script.

|  |  |  |
| --- | --- | --- |
|  | Name | Height |
|  | * Bob | 150 |
|  | * Jane | 127 |
|  | * Scott | 210 |
|  | * Drew | 197 |
|  | * Nancy | 110 |

We will pass these names in one at a time, then search for the user each time.

Graphical user interface, application

Description automatically generated Graphical user interface, text, application

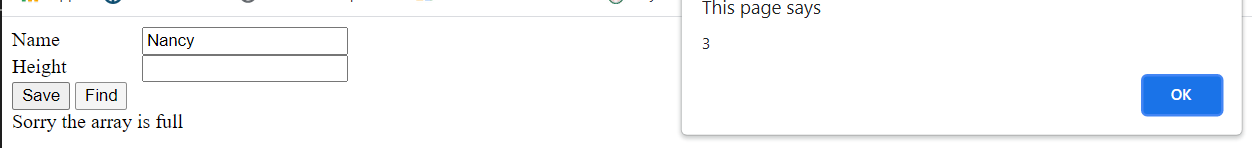
Description automatically generated Graphical user interface, text, application

Description automatically generatedTable

Description automatically generated

Graphical user interface, text, application

Description automatically generated



As you can see when we search for Nancy we get position 2 returned.

The names that we used in alphabetic order should be the follow:

Bob, Jane Drew, Nancy, Scott

When we take into account that the array position starts with 0 and count up we can see that Nancy is actually in position 3 of the array.

So from our testing we can see that the find function is performing correctly.

**Content Control:** Andrew Mills

## Content control

**Date**: 23/09/2021