

## Overview

The objective of this project is to practice the MIPS coding skills you have learned in the class. You will be implementing 2 algorithms in this project – selection sort and recursive summation.

## Description of `avgScore.s`

All the source code is contained in `avgScore.s`. This program asks the user for a series of assignment scores and stores them in an array. It then generates an array of sorted scores in descending order using **Selection Sort**. Next, it asks the user for the number of (lowest) scores to drop and calculates the sum of remaining scores using a **recursive algorithm**. Finally, it uses the sum to calculate the average score (that does not include the dropped scores). Your task is to implement the three (3) functions described below.

**Make sure you modify the main method ONLY in the indicated area (see comment in line 73)!**

To understand how arguments are passed to the following functions, study the `main` function **CAREFULLY**.

- **printArray**: This function prints the contents of an array. It takes in an array and its size as arguments. It does not return any value.
- **selSort**: This function performs **Selection Sort** in **descending order** on the array of scores. It takes in the length of the array as input. It populates the **sorted** array (defined in the data segment) with the contents of **orig** but in descending order. It does not return any value.
- **calcSum**: This function calculates the sum of an array's elements in a **recursive manner**. It takes in an array and its size as arguments. It returns the sum of elements in the argument array. You **must** implement this function **recursively**; therefore, be aware of the use of **stack memory**.

You may study `avgScore.c` to understand how your program should work. **Your output formatting must exactly match the sample run in terms of spacing, wording of prompts and newlines**. Create as many test cases as possible so that your program is free of error.

## Sample Run (user input shown in blue)

```
-----SAMPLE RUN 1
Enter the number of assignments (between 1 and 25): 5
Enter score: 2
Enter score: 22
Enter score: 11
Enter score: 7
Enter score: 19
Original scores: 2 22 11 7 19
Sorted scores (in descending order): 22 19 11 7 2
Enter the number of (lowest) scores to drop: 2
Average (rounded down) with dropped scores removed: 17
-- program is finished running --
```

## Collaboration

You must credit anyone you worked with in any of the following three different ways:

1. Given help to
2. Gotten help from
3. Collaborated with and worked together

## What to hand in

When you are done with this project assignment, submit all your work through CatCourses.

***Before*** you submit, make sure you have done the following:

- Your code compiles and runs on MARS.
- Attached `avgScores.s`.
- Attached a text document named `testRuns.txt` containing at least 5 test cases (see examples above).
- Filled in your collaborator's name (if any) in the "Comments..." textbox at the submission page.

Also, remember to demonstrate your code to the TA or instructor before the deadline.