It is what we think we know already
that often prevents us from learning.
~ *Claude Bernard*

CS1010 Programming Methodology

Week 12: Structures (Selected Answers)

---

5. Write a program **tiles.c** to read in an integer (greater than 1) indicating the number of tiles, followed by the tiles' data (length, width and price per square metre). A structure called **tile_t** should be created and the tiles' data should be stored in an array of such structure. The program then computes and outputs the difference in cost between the cheapest tile and the most expensive tile.

   (Actually, to get the answer there is no need to store the data in an array. This is done just for you to practise using array of structures.)

   The length and width are integers in metres, while the price, in dollars, is of type **float**. You may assume that there are at least 2 tiles and at most 20 tiles.

   You should write a modular program with the following functions:

```
// To read tiles' data into array tiles
// Return the number of tiles read
int scan_tiles(tile_t tiles[]);

// Return the difference in cost between cheapest
// tile and most expensive tile in the array tiles
float difference(tile_t tiles[], int size);
```

   A skeleton program **tiles_skeleton.c** is given. A sample run is shown below.

```
Enter number of tiles: 5
Enter data for 5 tiles:
5 8 0.20
3 5 0.18
6 10 0.31
4 6 0.27
2 4 0.38
Largest difference = $15.90
```

Answer: See **tiles/tiles.c**

6. **Cumulative Average Point (CAP)**

Write a program **cap.c** that makes use of these structures:

- **result_t** that contains 3 members: a 7-character module code, the grade obtained by the student, and the number of modular credits (MCs) of that module; and
- **student_t** that contains the student's name (at most 30 characters), and an array of result_t structures. You may assume that a student can take at most 50 modules.

Your program should read in a student's name, the number of modules he has taken, and for each module, the module code, the grade obtained, and the number of modular credits. All these data should be stored in a **student_t** variable. Your program should then compute the student's CAP, based on this formula:

CAP = Σ (MCs × Grade Point) / Σ (MCs)

The table below shows the grade point corresponding to each grade:

Grade	A+ or A	A-	B+	B	B-	C+	C	D+	D	F
Grade Point	5.0	4.5	4.0	3.5	3.0	2.5	2.0	1.5	1.0	0

For example, if Brusco Beh's has taken 5 modules and his results (module code, grade obtained, and number of MCs) are as follows:

```
CS1010 A+ 4
CS1231 B 4
MA1101R B+ 4
GEM1211 A- 3
PH2001 C 4
```

then his CAP is calculated as follows:

(5.0×4 + 3.5×4 + 4.0×4 + 4.5×3 + 2.0×4) / (4 + 4 + 4 + 3 + 4) = 71.5/19 = **3.76**

A skeleton program **cap_skeleton.c** is given. A sample run is shown below.

```
Enter student's name: Brusco Beh
Enter number of modules taken: 5
Enter results of 5 modules:
CS1010 A+ 4
CS1231 B 4
MA1101R B+ 4
GEM1211 A- 3
PH2001 C 4
CAP = 3.76
```

Answer: See **cap/cap.c**