TIC1002: Introduction to Computing and Programming II
Semester II, 2017/2018

Practical Exam 1 (24th February, 2018)

Task 1. Healthy Wheats [5 marks]

Old MacDonald decided to turn to farming after years of herding noisy animals. As a "scientific minded" person, he wanted to choose the healthiest wheats to plant in his new farm. His choosing criteria are quite straightforward: the wheat cannot be too tall (easily pick off by birds) or too short (not enough yield). After collecting a series of wheat height data, he approached you for help.

Implement a function where

`int within( int array[], int size, int lower, int upper, int result[]);`
array[] is an integer array with **size** number of data **lower** and **upper** are the value bounds, where **lower** <= **upper** **result[]** contains data from **array[]** that is within the bounds [**lower**…**upper**], inclusive. The relative order of the data should be the same as in the **array[]**.
This function returns the number of data that is within the bounds, i.e. the number of data in **result[]**.

Sample results

array[], size 6	lower	upper	Return result	result[]
{4, 1, 3, 1, 3, 2}	2	3	3	{3, 3, 2}
{4, 1, 3, 1, 3, 2}	1	4	6	{4, 1, 3, 1, 3, 2}
{4, 1, 3, 1, 3, 2}	5	10	0	{}

Note that we will check `result[]` to ensure no irrelevant data are stored.

Additional Requirements

Implementation	Maximum Mark
Iterative (used any form of loop)	**2 marks**
Recursive: Return the right result, but **result[]** is incorrect.	**3 marks**
Recursive: Return the right result and **result[]** is correct.	**5 marks**

Task 2. Lightspeed Travel Made Easy [15 marks**]

The recently promoted Starship Captain Burnam has a burning problem. With nearly depleted fuel reserve, her starship can only travel a certain distance before stopping for refueling. As her navigation chief, you are tasked to come up with a **list of suitable planets within certain distance from the starship location**.

Basic Configuration

The locations of the starship and planets are represented as **3 dimensional coordinates**. A structure with three fields is the best match in C:

```
struct Point3D {
  int X, Y, Z;   //note: capital letters
};
```

Implement a function

```
int rank_reachable(Point3D planets[], int size,
            Point3D* starship,
            double maxDistance, Point3D reachable[]);
```

planets[] is an array with **size** number of **Point3D** structures representing the location of planets.
starship is a **Point3D** structure indicating the current location of starship.
maxDistance is the maximum distance the starship can travel.

reachable[] is the resultant array which contains the planets that is **lesser than** the maximum distance from starship.

The results are sorted in order of distance from starship. If two planets P1 and P2 has the same distance from starship, they are ordered by X- followed by Y- and then Z-coordinates.

This function returns the number of reachable planet(s).

Sample Output

The following planets, stored in a **planets[]** array, are processed with respect to a **starship at coordinate (4, 4, 4).** Each planet is given a small letter name for **ease of reference only.**

	a	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*
X	-9	9	8	3	-4	5	-4	3	10	-3
Y	1	10	-3	3	-9	3	3	5	-3	-5
Z	0	10	-6	3	5	3	-3	3	-9	8
Dist	**13.92**	**9.84**	**12.84**	**1.73**	**15.29**	**1.73**	**10.67**	**1.73**	**15.93**	**12.08**

The function call:
`rank_reachable(planets, 10, &starship, 12, suitablePlanets)` returns 5 reachable planet sorted as follows in the `suitablePlanet[]` array:

	d	*h*	*f*	*b*	*g*
X	3	3	5	9	-4
Y	3	5	3	10	3
Z	3	3	3	10	-3
Dist	1.73	1.73	1.73	9.84	10.67

Pay attention to the ordering between planets ***d***, ***h*** and ***f***. As they are at equal distance from the starship, they were ordered based on X- coordinate, then by Y-, followed by Z- coordinates.

Restrictions:

- You are not allowed to modify the structure definition nor the function header for `rank_reachable()`.
- You **must** break the solution into meaningful **functions** instead of cramming all logic into the `rank_reachable()` function.

References:

a. The distance between two 3D points can be calculated as:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

where (X_1, Y_1, Z_1) and (X_2, Y_2, Z_2) are the coordinates of the two points respectively. You are free to use any built in math function to help.

b. An **integer bubble sort** is provided in the template code. You are free to copy and adapt the code as necessary. You are also **free to implement any sorting algorithm covered in the course** if you choose to

Rewarding Good Programming Style [4 marks]

4 marks out of the 15 marks are used for **programming style** check as follows:

Programming Style	Applicable to	Total
Consistent Indentation	Tasks 1 and 2	1 mark
Good variable naming	Tasks 1 and 2	1 mark
Modularization	Task 2 only	2 marks

~~~ End of PE Question ~~~