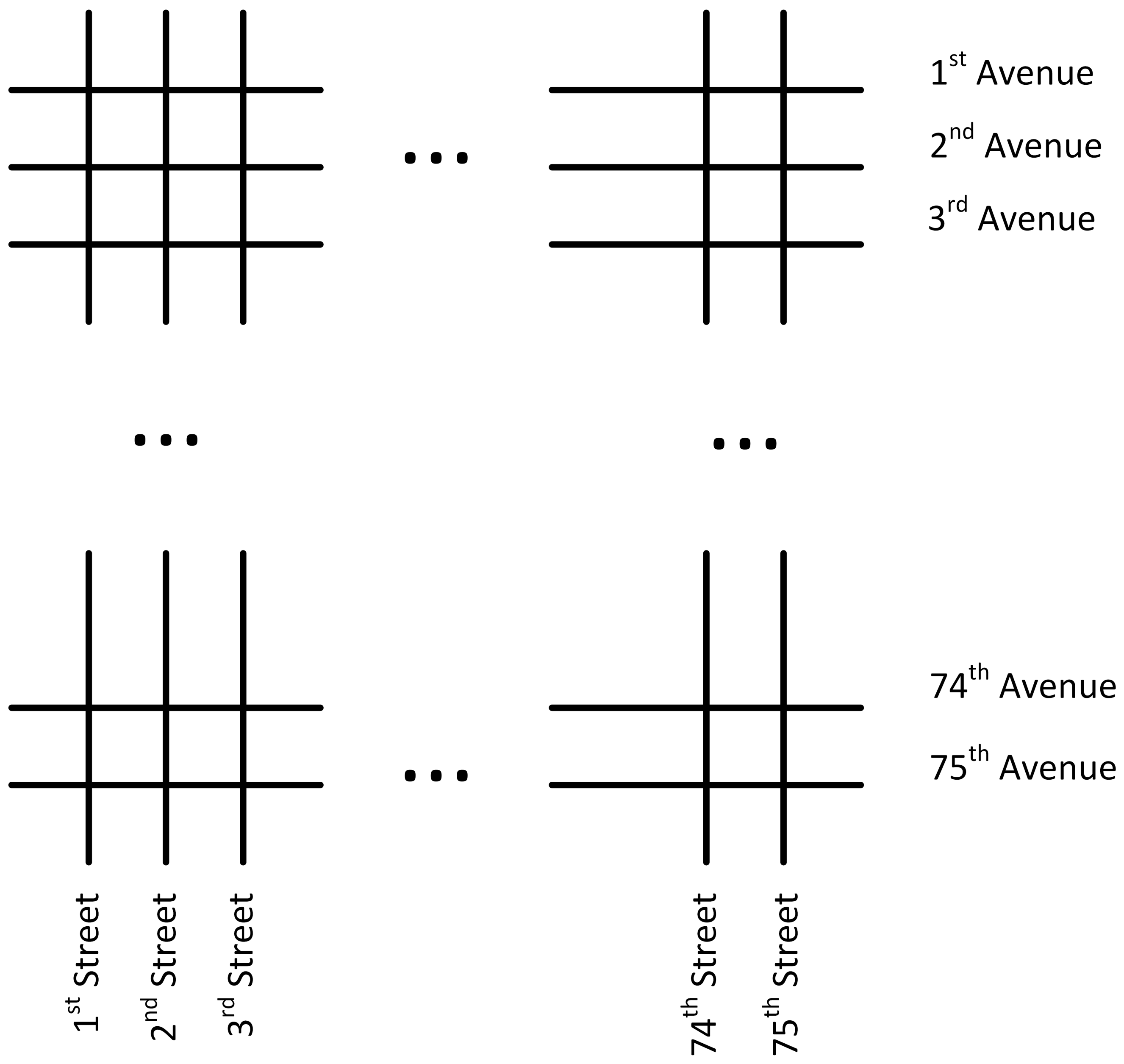
Problem Number: MME-PROB-10-03

Problem Title: Filling Potholes

Code filename: filling_potholes.cpp

This problem appeared as Question #4 on the GENE 121 Winter 2014 final exam.

Saajan has decided he is going to try and help cities determine how best to fill the potholes in their roads during the winter (and you have been asked to help). He has decided to start with one section of Calgary as a test because the roads are laid out in a grid pattern, as shown below. The city has 75 Streets and 75 Avenues.



During the winter, city staff record potholes in the file **potholes.txt**. Each row in the file lists the intersection (street and avenue) closest to the pothole, and the radius of the pothole in millimeters (which is assumed to be shaped as a hemisphere). All values are integers. The first three lines of the file are given as:

```
5        23       83
56       14       460
18       75       28
```

Near 5th St. and 23rd Ave., there is an 83 mm radius pothole. A very large pothole (nearly half a meter in radius) is located near 56th St. and 14th Ave.

The pothole information is to be stored in a 2D array. The first index refers to the street number and the second index refers to the avenue number.

Function - pothole_volume

Write a function called pothole_volume() that calculates the volume of a pothole in m^3. The function receives the radius of the pothole in millimeters.

The volume of a hemisphere is: V = (2/3) * PI * radius^3

Function - read_file

Write a function called read_file() that reads the input file into the pothole array. The function must receive the given information in the following order:

1. already opened input file
2. pothole array

Function - repair_volume

Write a function called repair_volume() that calculates the total amount of asphalt (in m^3) needed to repair a given street or avenue. This means filling in every pothole on the given street/avenue. This function must call the pothole_volume() function. The function must receive the given information in the following order:

1. pothole array
2. the number of the desired street/avenue
3. whether the given number refers to a street or not

   true : street

   false : avenue

Function - main

The main function should do the following:

- Opens the input file (potholes.txt) and verifies that it opened correctly.
- Declares an array to store the pothole data. Calls the read_file() function to read the data from the input file into the array.
- Finds and outputs the street or avenue that should be repaired next based on which street or avenue requires the closest to one bucket's worth of asphalt (0.25 m^3), without

going over.

Marmoset Testing Requirements

In order for your code to be evaluated by Marmoset, the main() function must be surrounded by the preprocessor directive MARMOSET_TESTING. Your other functions must not be contained within this preprocessor directive. Here is sample code:

```
// function declarations go here
#ifndef MARMOSET_TESTING
int main()
{
    ...
    return 0;
}
#endif
// function definitions go here
```

Sample Input

A sample potholes.txt file is provided.

Sample Output

The sample potholes.txt file should produce the following output:

```
The next road to repair is 56 Street.
```

Time Target

Time to Complete	Rating
Less than 45 minutes	* * *
45 to 60 minutes	* *
More than 60 minutes	*

© 2019 DAVID LAU ALL RIGHTS RESERVED