Sabanci University

Faculty of Engineering and Natural Sciences
CS204 Advanced Programming
Summer 2019-2020

Take-Home Exam 4 – Real Estate Management
Due: 17 August 2020 23:55 (SHARP)

DISCLAIMER:

Your program should be a robust one such that you have to consider all relevant user mistakes and extreme cases; you are expected to take actions accordingly!

Only checking the sample run cases might not be sufficient as your solution will be checked against a variety of samples different from the provided samples; however checking these test cases are highly encouraged and recommended, as well.

You must NOT **collaborate with your friends and discuss your solutions with each other. You have to write down the code on your own. Plagiarism will not be tolerated AND cooperation is not an excuse!**

After this take-home exam deadline, all students will be called for a demo to explain how they approached and solved the problem.

Introduction

The aim of this take-home exam (THE) is to have you work with multithreaded programming. You are asked to implement a basic variant of a machine learning algorithm in order to estimate the selling price of a house by analyzing a real data set.

Inputs and Files to Read

We have given you a real dataset from California (***`data.tsv`***). This dataset contains more than 20.000 districts (*Turkish*: mahalle) of California. Each district is given with a unique ID number, the median house age, average house room count, average house bedroom count, latitude and longitude. The first line of the file has these column names which you do not need to read. After this initial line, every other line has the information related to one district. There is a single tab (\t) character between each piece of information. In the very beginning of the program, *the whole file should be* ***read by your main thread*** *and these districts should be* ***stored in a queue***. You can read this file just like a .txt file with the methods you already know.

Once the file is read and loaded into the queue, the user should be asked for a series of inputs. First of all, we ask how many threads they want to use. As we have more than 20k districts to analyze, we need to distribute this computational load to multiple threads for efficiency in terms of speed. Therefore, our algorithm should be working on multiple threads and the number of threads is determined by the user. Afterwards, we will ask the features of the house the user is trying to sell. You can check the `Sample Runs` section for details on these inputs.

Task and Data Structure

Your task in this THE is to estimate the selling price of the given house. In Machine Learning, there are many sophisticated algorithms to be used in these scenarios. However, for the sake of simplicity, we will go with a very basic approach. Your program will deploy an algorithm which finds the 3 districts whose characteristics resemble our user's house the most. The average of the median house prices of these 3 districts will be our final prediction.

For finding how similar a district is to our user's house, we need to define a dissimilarity metric. The metric that you are going to use in this THE is the **square root** of the summation below:

```
(House Age Difference / 26)^2
+ (Room Count Difference / 10)^2
+ (Bedroom Count Difference / 4)^2
+ (Latitude Difference / 1)^2
+ (Longitude Difference / 1)^2
```

For example, if our user's house has the following information:

Age: 20, Rooms: 5, Bedrooms:2, Latitude: 30, Longitude: -120

And the candidate district has the following information:

Median House Age: 25, Average Rooms: 10, Average Bedrooms: 4, Latitude: 25, Longitude: -120

The dissimilarity score will be the **square root** of:

```
   (|20-25|/26)^2 + (|5-25|/10)^2 + (|2-10|/4)^2 + (|30-25|)^2 + (|-120
+120|)^2
```

The lower this score is, the more similar the district is to our user's house. Our ultimate goal is to find the 3 districts with the lowest results.

You may wonder where the divider numbers (26,10,4, 1, 1) come from. These are the numbers that show how significant a feature is. For example, 1 unit of change in latitude means 111km physical distance. Obviously, a building being one year older is less significant than 111km of neighbourhood change while predicting price change. These constants normally come from statistical analysis of the whole data set, but for this THE, we just give it to you as constants, in order not to complicate the THE. If you are curious, you can play with these constants to see how the judgement is affected by them. You can indeed find better constants than ours, but

submit your THE with the constants that we supply, which are (26,10,4, 1, 1).

The threads that are spawned by the main thread should be reading the districts from the queue (constructed by the main thread, as mentioned above) one by one and calculate the dissimilarity metric between the user's house and that district. Obviously, each district will be processed by a single thread and will be disposed of from the queue. While doing this, each thread should keep the 3 districts that have the lowest scores that it has seen, in a memory exclusive to that thread. You may think like "If we have three threads, each one can find the best one they come across and we get three best in the end". We are leaving this discussion to you to find why this proposal is incorrect :) But anyways, each thread should store the best three districts it came across. Be reminded that these threads ***should not*** be accessing the district queue at the same time or bad things will happen. You should utilize a ***mutex*** mechanism in order to make sure that read-write conflicts will not ever happen.

When a thread cannot find any more items in the district queue, it should write the best three dissimilarity scores to another queue (say the `result` queue) and exit. Again, this queue is shared between multiple threads as well; so the accesses to this queue should be protected by a mutex as well. When all the operations by the threads are done, there should be (`3*numThreads`) scores at the `result` queue. At this point, you may realize that storing the resulting scores is not enough; because the main thread that will get these scores would not know which district has which score. Due to this, this `result` queue should hold more than one piece of information. You may figure out what main thread will need in this result structure.

In your program, there should be two queues that store different kinds of information. These queues ***must be*** **dynamic queues**, just like the `DynIntQueue` that we taught you during lectures/labs. The first queue will store the information about the districts, while the second one is totally up to you, but it is probably going to be different than the first one. To tackle this problem, you may implement two different *DynQueue* classes with separate names and files; or, you may implement the *DynQueue* class in a templated way such that it can work with any type of data. Templated implementation is ***not*** a must, but it will be a cleaner solution and also will be a good chance of practice before the final exam.

After your threads **join** within the main thread, your main thread will calculate the absolute best three districts and report the information to the console. The average of the median prices of these three districts should also be calculated and printed.

During threads' calculations, we oblige you to print some intermediary information so that we observe the concurrency of your program. The threads should give an output on every 1000th item that they are processing, along with their thread ids. Thread id can be fetched by the following statement:

```
this_thrad::get_id()
```

A similar output should be given once the thread is finished with its work. Output format can be

observed in the Sample Runs. Be strictly reminded that, as console is a resource used by multiple threads at the same time, cout calls should be surrounded by a dedicated mutex.

Sample Runs

Below, we provide some sample runs of the program that you will develop. The *italic* and **bold** phrases are the standard input (cin) taken from the user (i.e., like ***this***). You have to display the required information in the same order.

We will not be using GradeChecker for this take-home exam.

The output format of your program should be exactly the same as sample runs. The resulting score values might be slightly different than the ones presented in the sample runs. For example: while the expected score of a district is `0.4917765`, your program may display `0.4917763`. This is natural, due to the fact that different computers have different CPUs and fractional numbers may lose precision during calculations. Your grade will not be affected by these very small differences. But, bigger calculation errors mean that your algorithm is defected, so you should fix them. Your thread ids and their ordering will also be different from these sample runs, do not worry about them.

Sample Run 1

```
20640 districts have been loaded.
How many threads?: 4

What is the age of the house?: 8
How many rooms does the house have?: 8
How many bedrooms does the house have?: 3
What is the latitude of the house?: 37.2
What is the longitude of the house?: -121.8

Thread with id 2456 is processing its 1000th item.
Thread with id 6596 is processing its 1000th item.
Thread with id 5252 is processing its 1000th item.
Thread with id 9160 is processing its 1000th item.
Thread with id 2456 is processing its 2000th item.
Thread with id 6596 is processing its 2000th item.
Thread with id 5252 is processing its 2000th item.
Thread with id 9160 is processing its 2000th item.
Thread with id 2456 is processing its 3000th item.
Thread with id 6596 is processing its 3000th item.
Thread with id 5252 is processing its 3000th item.
Thread with id 9160 is processing its 3000th item.
```

```
Thread with id 2456 is processing its 4000th item.
Thread with id 6596 is processing its 4000th item.
Thread with id 5252 is processing its 4000th item.
Thread with id 9160 is processing its 4000th item.
Thread with id 2456 is processing its 5000th item.
Thread with id 6596 is processing its 5000th item.
Thread with id 5252 is processing its 5000th item.
Thread with id 9160 is processing its 5000th item.
Thread with id 2456 is exiting.
Thread with id 6596 is exiting.
Thread with id 5252 is exiting.
Thread with id 9160 is exiting.

Closest three districts in terms of features have the following median
prices:

Closest District 1
--------
Id: 18503
Median Price: 390000
Calculated Dissimilarity Metric: 0.437829

Closest District 2
--------
Id: 17878
Median Price: 137500
Calculated Dissimilarity Metric: 0.44429

Closest District 3
--------
Id: 17745
Median Price: 335500
Calculated Dissimilarity Metric: 0.49071

Average median price of three closest districts: 287667
```

Sample Run 2

```
20640 districts have been loaded.
How many threads?: 8

What is the age of the house?: 45
How many rooms does the house have?: 8
How many bedrooms does the house have?: 3
```

```
What is the latitude of the house?: 38.5
What is the longitude of the house?: -122.9

Thread with id 2760 is processing its 1000th item.
Thread with id 932 is processing its 1000th item.
Thread with id 12704 is processing its 1000th item.
Thread with id 10164 is processing its 1000th item.
Thread with id 12316 is processing its 1000th item.
Thread with id 3864 is processing its 1000th item.
Thread with id 7464 is processing its 1000th item.
Thread with id 4772 is processing its 1000th item.
Thread with id 2760 is processing its 2000th item.
Thread with id 932 is processing its 2000th item.
Thread with id 12704 is processing its 2000th item.
Thread with id 10164 is processing its 2000th item.
Thread with id 12316 is processing its 2000th item.
Thread with id 3864 is processing its 2000th item.
Thread with id 7464 is processing its 2000th item.
Thread with id 4772 is processing its 2000th item.
Thread with id 2760 is processing its 3000th item.
Thread with id 932 is processing its 3000th item.
Thread with id 12704 is processing its 3000th item.
Thread with id 10164 is processing its 3000th item.
Thread with id 12316 is exiting.
Thread with id 3864 is exiting.
Thread with id 7464 is exiting.
Thread with id 4772 is exiting.
Thread with id 2760 is exiting.
Thread with id 932 is exiting.
Thread with id 12704 is exiting.
Thread with id 10164 is exiting.

Closest three districts in terms of features have the following median
prices:

Closest District 1
--------
Id: 19323
Median Price: 132700
Calculated Dissimilarity Metric: 0.37901

Closest District 2
--------
Id: 19318
Median Price: 120000
```

```
Calculated Dissimilarity Metric: 0.401863

Closest District 3
--------
Id: 19331
Median Price: 117100
Calculated Dissimilarity Metric: 0.4035

Average median price of three closest districts: 123267
Press any key to continue . . .
```

Sample Run 3

```
20640 districts have been loaded.
How many threads?: 2

What is the age of the house?: 18
How many rooms does the house have?: 5
How many bedrooms does the house have?: 1
What is the latitude of the house?: 33.9
What is the longitude of the house?: -117.93

Thread with id 12528 is processing its 1000th item.
Thread with id 5296 is processing its 1000th item.
Thread with id 12528 is processing its 2000th item.
Thread with id 5296 is processing its 2000th item.
Thread with id 12528 is processing its 3000th item.
Thread with id 5296 is processing its 3000th item.
Thread with id 12528 is processing its 4000th item.
Thread with id 5296 is processing its 4000th item.
Thread with id 12528 is processing its 5000th item.
Thread with id 5296 is processing its 5000th item.
Thread with id 12528 is processing its 6000th item.
Thread with id 5296 is processing its 6000th item.
Thread with id 12528 is processing its 7000th item.
Thread with id 5296 is processing its 7000th item.
Thread with id 12528 is processing its 8000th item.
Thread with id 5296 is processing its 8000th item.
Thread with id 12528 is processing its 9000th item.
Thread with id 5296 is processing its 9000th item.
Thread with id 12528 is processing its 10000th item.
Thread with id 5296 is processing its 10000th item.
Thread with id 12528 is exiting.

Closest three districts in terms of features have the following median
```

```
prices:

Closest District 1
--------
Id: 10143
Median Price: 193500
Calculated Dissimilarity Metric: 0.0643439

Closest District 2
--------
Id: 10105
Median Price: 153600
Calculated Dissimilarity Metric: 0.0672495

Closest District 3
--------
Id: 11698
Median Price: 181800
Calculated Dissimilarity Metric: 0.0728733

Average median price of three closest districts: 176300
Press any key to continue . . .
```

Sample Run 4

```
20640 districts have been loaded.
How many threads?: 16

What is the age of the house?: 18
How many rooms does the house have?: 5
How many bedrooms does the house have?: 1
What is the latitude of the house?: 33.9
What is the longitude of the house?: -117.93

Thread with id 3356 is processing its 1000th item.
Thread with id 13036 is processing its 1000th item.
Thread with id 12692 is processing its 1000th item.
Thread with id 6172 is processing its 1000th item.
Thread with id 11940 is processing its 1000th item.
Thread with id 12272 is processing its 1000th item.
Thread with id 9924 is processing its 1000th item.
Thread with id 2276 is processing its 1000th item.
Thread with id 10960 is processing its 1000th item.
Thread with id 11368 is processing its 1000th item.
Thread with id 6624 is processing its 1000th item.
Thread with id 12568 is processing its 1000th item.
Thread with id 13304 is processing its 1000th item.
```

```
Thread with id 12012 is processing its 1000th item.
Thread with id 6736 is processing its 1000th item.
Thread with id 9180 is processing its 1000th item.
Thread with id 3356 is processing its 2000th item.
Thread with id 13036 is processing its 2000th item.
Thread with id 12692 is processing its 2000th item.
Thread with id 6172 is processing its 2000th item.
Thread with id 11940 is exiting.
Thread with id 12272 is exiting.
Thread with id 9924 is exiting.
Thread with id 2276 is exiting.
Thread with id 10960 is exiting.
Thread with id 11368 is exiting.
Thread with id 6624 is exiting.
Thread with id 12568 is exiting.
Thread with id 13304 is exiting.
Thread with id 12012 is exiting.
Thread with id 6736 is exiting.
Thread with id 9180 is exiting.
Thread with id 3356 is exiting.
Thread with id 13036 is exiting.
Thread with id 12692 is exiting.
Thread with id 6172 is exiting.

Closest three districts in terms of features have the following median
prices:

Closest District 1
--------
Id: 10143
Median Price: 193500
Calculated Dissimilarity Metric: 0.0643439

Closest District 2
--------
Id: 10105
Median Price: 153600
Calculated Dissimilarity Metric: 0.0672495

Closest District 3
--------
Id: 11698
Median Price: 181800
Calculated Dissimilarity Metric: 0.0728733

Average median price of three closest districts: 176300
Press any key to continue . . .
```

Some Important Rules

Although some of the information is given below, please also read the take-home exam submission and grading policies from the lecture notes of the first week. In order to get a full credit, your program must be efficient, modular (with the use of functions), well commented and indented. Besides, you also have to use understandable identifier names. Presence of any redundant computation, bad indentation, meaningless identifiers or missing/irrelevant comments may decrease your grade in case that we detect them.

When we grade your take-home exams, we pay attention to these issues. Moreover, in order to observe the real performance of your code, we are going to run your programs in Release mode and **we may test your programs with even larger test cases**. Hence, take into consideration the efficiency of your algorithms other than correctness.

How to get help?

You may ask your questions to TAs or to the instructor. Information regarding the office hours of the TAs and the instructor are available at SUCourse+.

YOU CANNOT USE GRADE CHECKER FOR THIS TAKE-HOME EXAM!

We will *not* be using GradeChecker for this take-home exam. As this program has a big dataset, the task is multi-threaded and floating point operations suffer precision in computers, using an automated tool would not be a good choice.

Therefore, **your THE will be graded manually. Misuse of multi-threading or bad practices will be inspected manually and result in serious grade deductions. *If you implement the given algorithm in a single-threaded fashion, you will get 0*. If threads block each other much and performance suffers because of this, your grade will be hurt badly.**

The internal clock of SUCourse+ might be a couple of minutes skewed, so make sure you do not leave the submission to the last minute. In the case of failing to submit your take-home exam on time:

"No successful submission on SUCourse+ on time = A grade of 0 directly."

What and where to submit (PLEASE READ, IMPORTANT)

It'd be a good idea to write your name and lastname in the program (as a comment line of course). Do not use any Turkish characters anywhere in your code (not even in comment parts).

Submission guidelines are below. If you do not follow these guidelines, your grade will be *zero*.

- Name your cpp file that contains your program as follows:

  "SUCourse+UserName_the4.cpp"

  Your SUCourse+ username is actually your SUNet username which is used for checking sabanciuniv e-mails. Do NOT use any spaces, non-ASCII and Turkish characters in the file name. For example, if your SU e-mail address is **atam@sabanciuniv.edu**, then the file name must be: **"atam_the4.cpp"**

- Please make sure that this file is the latest version of your take-home exam program.

- You should upload `data.tsv` to SUCourse+ as well,

- Do not zip any of the documents. Upload all of them as separate files.

- Submit your work **through SUCourse+ only**! You can use the GradeChecker only to see if your program can produce the correct outputs both in the correct order and in the correct format. It will not be considered as the official submission. You must submit your work to SUCourse+.

- If you would like to resubmit your work, you should first remove the existing file(s). This step is very important. If you do not delete the old file(s), we will receive both files and the old one may be graded.

You may visit the office hours if you have any questions regarding submissions.

Plagiarism

Plagiarism is checked by automated tools and we are very capable of detecting such cases. Be careful with that...

Exchange of abstract ideas are totally okay but once you start sharing the code with each other, it is very probable to get caught by plagiarism. So, do NOT send any part of your code to your friends by any means or you might be charged as well, although you have done your take-home exam by yourself. Take-home exams are to be done personally and you have to submit your own work. **Cooperation will NOT be counted as an excuse.**

In case of plagiarism, the rules on the Syllabus apply.

Good Luck!
Tolga Atam, Duygu K. Altop