Extreme Computing
Assignment 1

1 Introduction

This is the coursework assignment for the Extreme Computing course 2021/22. You need to use the Scala Collection API to solve problems you might encounter when working with collections. This section will give you administrative information and help with solving the assignment. This is followed by the actual tasks and finally a description of how to submit your solutions.

1.1 Administrative Information

Deadline The assignment is due at **4:00pm on November 1st**.

Deadline Extension The School of Informatics has a policy on coursework deadlines, which applies across all taught courses. Further information can be found here:

`http://web.inf.ed.ac.uk/infweb/student-services/ito/admin/coursework-projects/late-coursework-extension-requests`

Questions All questions should go on Piazza

`https://piazza.com/class/ktlmf55uoar593`

in the "hw1" folder. Feel free to discuss general techniques amongst each other unless you would reveal an answer. If your question / discussion reveals an answer, ask privately.

Marking The assignment is worth 50 marks in total and counts for 20% of the final course mark. Marks are given for correctness, efficiency and proper use of tools.

Marks are indicated on the right margin of the page by two numbers, e.g. **1+3 marks**. The first number indicates achievable marks for correctness while the second number indicates achievable marks for efficiency.

Submission The submission process for your solution is described in Section 4. We start marking at the deadline and only mark once. If you are submitting on time, you can submit as many times as you want and the last one will be marked. If you are submitting late, you may only submit once in total (which implies that you should not submit before the deadline) or run the risk that we will mark an old version then penalise for the last submission time.

Marking Feedback You will receive your marks and feedback for your solution on LEARN. Once you have received your marks, you have three days to ask questions about them, e.g. feedback clarification. Please post any such questions in a private Piazza message.

Good Scholarly Practice Please remember the University requirement as regards all assessed work for credit. Details about this can be found at:

> http://web.inf.ed.ac.uk/infweb/admin/policies/academic-misconduct

Furthermore, you are required to take reasonable measures to protect your assessed work from unauthorised access. For example, if you put any such work on a public repository then you must set access permissions appropriately (generally permitting access only to yourself, or your group in the case of group practicals).

2 Tasks

For the dataset you will use in this assignment, there is a `imdb-small-data.zip` file that can be accessed at:

https://amirsh.github.io/files/exc21/imdb-small-data.zip

You can copy the extracted `tsv` files to `src/main/resources/imdb/` for local testing and debugging.

2.1 Internet Movie Database (IMDB)

This assignment will be on processing the IMDB dataset – we have chosen a subset for these tasks to encourage you to think about how to structure your solutions to use multiple input data collections, and efficiently process structured text using Scala collections.

Please note that we have removed the first line of the `tsv` file, which contained the column names in the original dataset. We have done this for your convenience, as in your code, you can assume all lines are data. The four files used, including their structures, are detailed in Section 2.2.

Note that not all `.tsv` files are required for all questions. Consult the schema in Section 2.2 to ascertain which one(s) you require for the task at hand. Be aware that `skipVal` (`'\N'`) may be present where fields are denoted `Option`, meaning no data is present. You are expected to account for this possibility and ignore those entries from your solutions.

2.2 IMDB Schema Reference

The following table defines the columns in each of the provided files from the IMDB dataset to aid you in your solution design.

- `Option[T]` means either type `T` is present, or `skipVal ('\N')` otherwise
- `List[T]` means a comma-delimited list of type `T` is present, e.g. ‘`dog,cat,bear`’, where `T := String`

	Index	Field	Type	Examples/Notes
演员编号				**name.basics.tsv**
名字	0	nconst	String	nmXXXXXXX – Unique person/crew ID
	1	primaryName	Option[String]	–
	2	birthYear	Option[Int]	–
	3	deathYear	Option[Int]	–
	4	primaryProfession	Option[List[String]]	‘editor,manager’, ‘actor’, ‘actress’
	5	knownForTitles	Option[List[String]]	‘tconst1,tconst2,tconst3’
电影编号			task4	**title.basics.tsv**
细节	0	tconst	String	ttXXXXXXX – Unique title ID
	1	titleType	Option[String]	‘tvMovie’, ‘short’, ‘movie’, ‘videoGame’
	2	primaryTitle	Option[String]	–
	3	originalTitle	Option[String]	–
	4	isAdult	Int	– (YYYY-1900)/10
	5	startYear	Option[Int]	YYYY – Release year 1900-1999
	6	endYear	Option[Int]	YYYY – End year, e.g. when a play ends.
	7	runtimeMinutes	Option[Int]	–
	8	genres	Option[List[String]]	‘Documentary,Short,Sport’
电影编号				**title.crew.tsv**
导演	0	tconst	String	Joins title.basics.tconst
作家	1	directors	Option[List[String]]	‘nmXXXXXX1,nmXXXXXX2’ – Joins nconst
	2	writers	Option[List[String]]	‘nmXXXXXX1,nmXXXXXX2’ – Joins nconst
电影编号				**title.ratings.tsv**
评分	0	tconst	String	Joins title.basics.tconst
票数	1	averageRating	Float	–
	2	numVotes	Int	–

3 Tasks

Download imdb-src.zip and it extract it somewhere on your machine. You have to complete the missing implementations (specified by ???) in src/main/scala/imdb/ImdbAnalysis.scala.

You are encouraged to look at the Scala API documentation while solving this exercise, which can be found here:

https://www.scala-lang.org/api/2.12.15/index.html

Consult the schema in Section 2.2 when designing your solutions in order to extract the correct data.

666

Task 1

◁ Task

Calculate the average, minimum, and maximum runtime duration for all titles per movie genre.

Note that a title can have more than one genre, thus it should be considered for all of them. The results should be kept in minutes and titles with 0 runtime duration are valid and should be accounted for in your solution. **(5+6 marks)**

return type: List[(Float, Int, Int, String)]
avg_runtime:Float
min_runtime:Int
max_runtime:Int
genre:String

666

Task 2

◁ Task

Return the titles of the movies which were released between 1990 and 2018 (inclusive), have an average rating of 7.5 or more, and have received 500000 votes or more.

For the titles use the primaryTitle field and account only for entries whose titleType is 'movie'. **(6+7 marks)**

return type: List[String]
title:String

Task 3

◁ Task

Return the top rated movie of each genre for each decade between 1900 and 1999.

For the titles use the primaryTitle field and account only for entries whose titleType is 'movie'. For calculating the top rated movies use the averageRating field and for the release year use the startYear field.

The output should be sorted by decade and then by genre. For the movies with the same rating and of the same decade, print only the one with the title that comes first alphabetically. Each decade should be represented with a single digit, starting with 0 corresponding to 1900-1909. **(6+7 marks)**

return type: List[(Int, String, String)] 0 1 2 3 4 5 6 7 8 9
decade:Int
genre:String
title:String

Task 4

◁ Task

In this task we are interested in all the crew names (`primaryName`) for whom there are at least two known-for films released since the year 2010 up to and including the current year (2021). You need to return the crew name and the number of such films. **(6+7 marks)**

```
return type: List[(String, Int)]
crew_name:String
film_count:Int
```

4 Submissions

To submit your work, please do the following:

1. To test the correctness of your program, you can use the `test` command of sbt. We will run your submissions against a bigger test suite than we provided to you. So if you want full points, be sure to add your own tests to double check that you have caught all corner cases. We also check the performance of your implementation. Thus, make sure that you use the best possible implementation (e.g., `groupBy` instead of inefficient nested loops where possible).

2. To submit your code, please zip the entire `imdb` directory and name it as `imdb.zip`. Make sure that the data files are removed from the `resource` directory.

3. Upload the zip file to Learn.