University of Negros Occidental-Recoletos



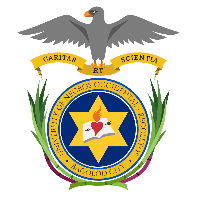
**Section:K11 and K12**

**CIT15223Z: Program Logic Formulation & Computer Programming 2**

**PERMUTATION**

**ENDTERM PROJECT**

**COLLEGE OF INFORMATION TECHNOLOGY – ITE - CODE**



**INSTRUCTIONS:**

* You must complete ALL the requirements.
* Marks will be awarded for good presentation and thoroughness in your approach.
* NO marks would be awarded for the entire project if any part is found to be copied directly from printed materials or from another student.
* Complete this cover sheet and attach it to your project.

**Students Declaration:**

I/We declare that:

* I/We understand what is meant by plagiarism (illegal copying of one’s work)
* The implication of plagiarism is tantamount to cheating
* This project is all my/our own work and I/we have acknowledged any use of the published and unpublished works of other people.

**Students’ Signature:**…………………………………………………………………………... **Date:** MARCH 28,2020

|  |  |  |  |
| --- | --- | --- | --- |
| Total number of pages including this cover page : 13 | | |  |
| Class Section | **K11** | Due Date | **March 30, 2020** |
| Student’s Course & Year | CABALAN,HURLY ZADE CHRSITIAN  ENRIQUEZ,ADRIAN | Submission Date | **March 28, 2020** |
| Students’Full Name |  | | |
| Instructors’Name | **ARIANE M. TORRES, MIT** | | |

FOR OFFICIAL USE ONLY

|  |  |  |  |
| --- | --- | --- | --- |
| Instructor’s/Marker’s Comments | | | |
|  | | | |
| Marker’s Name |  | Marks Awarded | / |
| Signature |  | Grade Equivalent |  |

1. **Introduction**

In our field, information technology, the thing that we usually do is turn abstract and complex notions into something concrete, tangible and useful. In simpler terms, we turn something that doesn’t make sense into something that does and make it relevant to our lives. We mainly translate these concepts into understandable language using Discrete Mathematics. Discrete mathematics allows us to understand relations between objects (set theory), how objects can be arranged and structured (combinatorics), understand how we come up with solutions (mathematical induction), etc. Discrete mathematics is essentially real life turned into numbers and as an IT student, is something we try to explore and utilize in everything that we do.

This project deals with permutation with and without repetition. Permutation is the number of ways a set of objects can be arranged. The permutation of a set of objects grows larger as its size increases. It is solved by simply getting the factorial of the number of objects in a set. If there are objects that are repeated, you take the factorial of each object’s frequency. This is done so that each repeated character would be treated as one unit. You then multiply each object’s frequency’s factorial, and divide it to the factorial of the size of the set. The number you’d get is then number of ways the set of objects can be arranged.

This application allows you to solve the permutation of a given input and shows you the solution on how it came up with the answer. It acts similar to a calculator but for permutation. It takes one input and shows how to get the number of permutations of the given input. It does not show the different forms it can take, only the number since the higher the length of the input, the more permutations it has and it tends to grow very large per additional character. Similar to most modern calculators, it also records the different words you have asked the application to solve and can be accessed again in a different page, solution history. The different entries can also be deleted in the solution history page can also be deleted. Even if you close the application, these records would still be available.

1. **Analysis and Design**

The features of the program endeavors to meet all these requirements: use permutation with and without repetition, have a main menu, use exception handling, allows user to add or remove data, using file inputs for data entry, and all of these using a JavaFX application.

|  |
| --- |
| Controller |
| -solutionWords: ArrayList<String>  - solutionsDone: ArrayList<String>  - soltnInputTextField: TextField  - soltnExplanationLabel: Label  - solutionHistoryGridPane: GridPane |
| + initialize(): Void  + showReference(ActionEvent event): Void  + showSolutionAndRecordIt(ActionEvent event): Void  - getParentFromFXML(String FXMLFilename): Parent  + gotoSolvePage(ActionEvent event): Void  + gotoMainMenu(ActionEvent event): Void  + gotoSolutionHistoryPage(ActionEvent event): Void  - showHistoryInGridPane(): Void  - changePage(String FXMLSceneName, ActionEvent event): Void  - getCurrentStageFromEvent(ActionEvent event): Stage  - addSolutionToArrayList(String solution): Void  - writeAllToFile(): Void  - getAllDataFromTextFile(): String[]  - populateArrayList(String[] historyData): Void  - writeSolutions(FileWriter solutionWriter): Void  - getPermutationWithRepetition(String characters): double  - explanationOfPermutationSolution(String characters): String  - hasRepetition(String characters): Boolean  - getFactorial(int number): int  -stringIsNull(String string) :Boolean  - countRepetition(String characters): LinkedHashMap<Characters, Integers> |

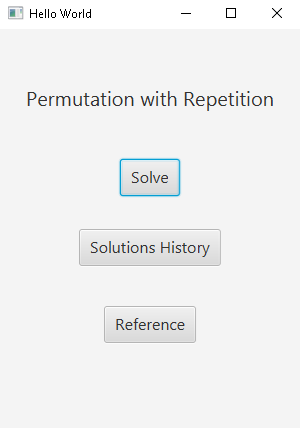
|  |  |  |
| --- | --- | --- |
| **Input** | **Process** | **Output** |
| **Program startup (initialize)** | **-if textfile not empty, takes all of its data and adds them to arraylist**  **-populates “SolutionHistoryPage.fxml” scene with this data** | **-Shows main menu**  **-Buttons created in “SolutionHistoryPage.fxml”** |
| **Click Solve on Main Menu** | **-Changes stage’s scene to “SovePage.fxml”** | **-shows scene that allows user to solve permutation data with characters input.** |
| **-Type “arim” on textfield**  **-click “Find Permutation” button** | **-takes textfield input and removes all spaces and changes it to uppercase.**  **-check if input only contains spaces or empty, then informs user else, shows solution.**  **-changes Label text to solution**  **-ArrayList, textfile, “SolutionHistoryPage” gets updated with this new data** | **-Shows the permutation and how its solved given user input.** |
| **Back button clicked** | **-Changes scene back to main menu** | **-shows main menu filled with button to navigate through different pages.** |
| **Click “Solution History” button** | **-Changes Stage scene to “SolutionHistoryPage.fxml”** | **-shows scene with buttons of previously solved words in “SolvePage”** |
| **Click button for word that was solved** | **-Creates scene that contains the solution for the word and changes to it.**  **-back button also created** | **-Shows the solution for the word.** |
| **Click delete button** | **-removes the data from the ArrayList**  **-updates textfile with new arraylist data**  **-refreshes page with new data** | **-Removes button for the particular word’s solution** |

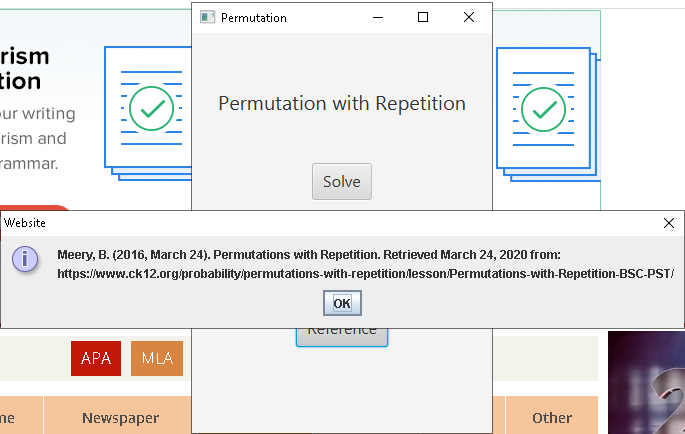
1. **Development**

During this community quarantine, our group decided to divide the tasks based on our available resources and materials on our respective houses. The tasks are divided into two: The coding and actual program project (CABALAN, 1st group) and the documentation for the project (ENRIQUEZ, 2nd group). In order to establish a good and harmonic output, since we know that it is difficult to do documentation without having a finished program, the first group (coding and project program) continuously updates the other group (documentation) on the development of the project. This is outlined below.

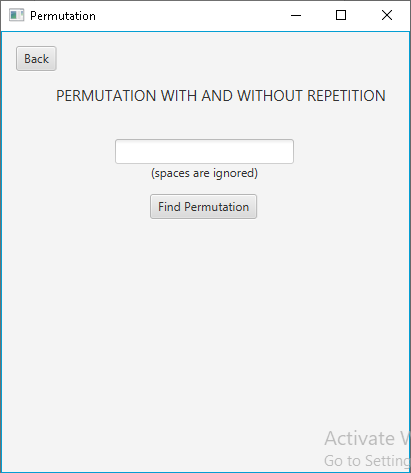
|  |  |  |
| --- | --- | --- |
| Day | Tasks | Solution |
| Day 1 (Sunday)  March 22, 2020 | Choosing the concept of the project.  Documentation. | Choosing from the concept provided in the final project pdf. We chose Permutation as our project for we think that learning more on this topic would be useful eventually in our field(i.e. security and password).  The 2nd group makes the introduction. 1st group starts setting up technicalities. |
| Day 2 (Monday)  March 23, 2020 | Creating scenes and planning the code.  Start of coding. Documentation. | Simple layout of the UI done by the 1st group.  Send screenshots to the 2nd group for documentation. |
| Day 3 (Tuesday)  March 24, 2020 | Continuation of coding process based on the requirements provided. Documentation. | The implementation exception handling, file handling etc., by 1st group.  2nd group updates IPO of the documentation and makes course feedback and everything else in the meantime. |
| Day 4 (Wednesday)  March 25, 2020 | Coverting into Jar file. Struggle in converting to Jar file. Documentation must finished. | Searching for posible solurion using Youtube and Google. Final polish of the documentation if the requirements were met. |
| Day 5  March 26, 2020 | Got an update, jar not required; ditched it completely.  Bug found. | 1st group works on debugging  2nd group fixes documentation accordingly |
| Day 6  March 27,2020 | Bug Fixed  Documentation finalization | 1st group fixes the bug  2nd group finalizes documentation, and proofreads everything. |

1. **Program Screenshots**

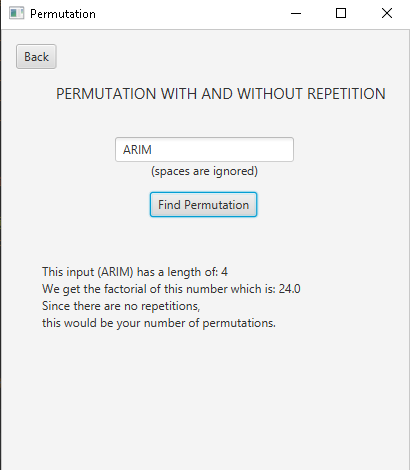
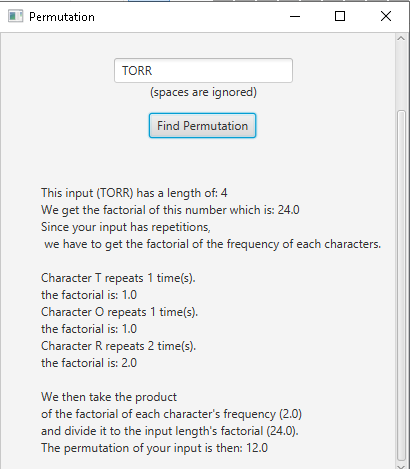
****



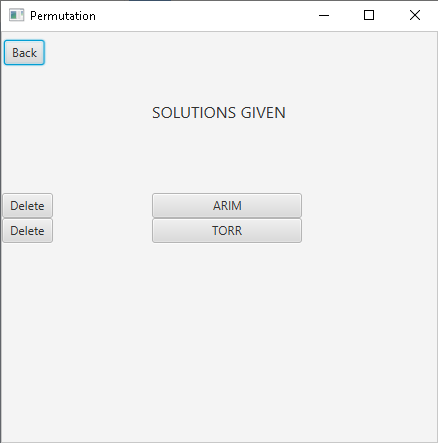
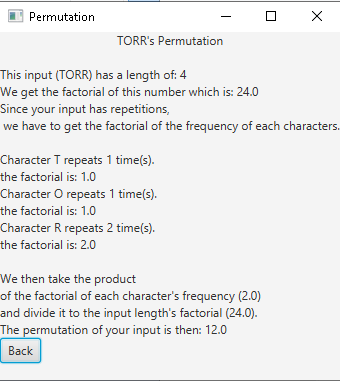
This is the main menu of the application. This page is where the user will navigate through the different sections of the application. Clicking “Solve” will redirect the user to a scene where he can give an input and solve its permutation. Clicking “Solutions History” will show the user all of the input that was solved and view it again. Clicking “Reference” will show all references used for the creation of this application through a JOPtionPane.



This is the scene you get redirected to when the user clicks the “Solve” button on the main menu. The user can put an input and then the application will use that input to solve its permutation. If the user clicks on the button while the text field is empty or if it only contains spaces, the application will warn the user accordingly.



This is a sample run of the application. The application performs additional operations if there are repetitions in the user’s input and calculates the permutation accordingly. It also shows the steps in how it was able to reach the solution.



This is the scene the user gets redirected to if he clicks on the “Solution History” button on the main menu. This is where all the input that has been solved can be reviewed by the user. The user also has the option to delete it if it becomes too populated. Clicking on one of the input that was solved will redirect the user to a scene that shows the solution in getting its permutation similar to the one the user receives at the “Solve” page.

1. **Course Feedback**

Learning must be a mission of an individual in his/her life. As a future professional in the field of computer science and information technology, we are building and collecting pieces of passion and skills with the help of our teachers and colleagues to be a successful professional someday.

On our subject CIT152 (Programming and Logic formulation 2) another programming language was introduced to us, the JAVA programming language which is class-based, object-oriented programming language. Unlike Python, Java require some extra lines first before you can begin actual coding. 2 lines of Python code is almost equal to 10 lines of Java. We have learned about: the use of methods and creating a method, calling of class methods using objects, method overloading, creating a class diagram of a class, recursion which is something like a loop but completely different, matrices and its application, using Java Swing to create a Graphical User Interface, file handling, exception handling, using JavaFX, in which Java swing can be implemented, data structures in Java and data types in java and we applied these concepts (e.i. summations, loops, user defined inputs, indices etc.) in our programs and applications. Loving what we have learned is like learning to love the field we've chosen, embracing its complexity and beauty to its core, taking risk to learn more and push forward not just to get the degree that we want but to gain learnings that will surely be useful in our future.

In every session from the subject Discrete math, I've learned a lot of useful concepts that are related to programming such as matrices, sets --which is like arrays--, permutation and so on. There are some topics that we had already taken during our senior high but it was interesting to see new ways in tackling these same problems that we had encountered in our previous years and learning more about it. There were also topics that was hard for us but we took it as a challenge to further our knowledge and made it more bearable by thinking of how it would be beneficial for us. We learned that the more we spend time learning it, the more it was willing to lend us its secrets. Yes, we learned a lot in discrete mathematics but this is not just what we have learned. We've also learned to give importance to mathematics because it is not only useful in programming but also in our day to day lives.

1. **References**

Meery, B. (2016, March 24). Permutations with Repetition. Retrieved March 24, 2020 from:\nhttps://www.ck12.org/probability/permutations-with-repetition/lesson/Permutations-with-Repetition-BSC-PST/"