COL732: Virtualization and Cloud Computing
Semester I, 2022-2023

Lab-1: Introduction to Rust

3 August 2022

Deadline: 20 August 2022

Submission Instructions

1. You will submit the source code in **zip** format to Moodle (Lab 1). The naming convention of the zip file should be <Entry_Number>_<First_Name>.zip. The starter code is available here.
2. The Lab would be **auto-graded**. Therefore, **follow** the same naming conventions described in the Deliverables section. Failing to adhere to these conventions will lead to zero marks in the Lab.
3. You should write the code **without** taking help from your peers or referring to online resources except for Rust documentation. Not doing any of these will be considered a breach of the honor code, and the consequences would range from zero marks in the Lab to a disciplinary committee action.
4. You can use **Piazza** for any queries related to the Lab.

Setup Instructions [1]

Rust installation procedure in Linux:

```
curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh
source "$HOME/.cargo/env" # In case you are unable to execute the rust command
```

Problem Statement

Solve the following problem statements in Rust:

1. (2 marks) Given an array of intervals, merge the overlapping intervals. *[Vector]*

   ```
   Sample Command: ./vector.out 1 9 7 10 13 18
   Sample Output: [Interval {start: 1, end: 10 }, Interval {start:
   13, end: 18 }]
   ```

2. (2 marks) Given two strings *s* and *t*, determine if they are isomorphic. Two strings are isomorphic if characters in *s* can be replaced to get *t* while preserving the order of characters, and the mapping between the characters must be one-to-one. *[Map]*

   ```
   Sample Command: ./map.out mice cuba
   Sample Output: true
   ```

3. (2 marks) Given a set of integers, print all possible subsets (powerset). *[Recursion]*

   ```
   Sample Command: ./recursion.out 1 2
   Sample Output: [[], [2], [1], [1, 2]]
   ```

4. (2 marks) Given two non-negative integers having same number of digits, num1 and num2 represented as strings, return the product of num1 and num2, also represented as a string. *[Type Cast]*

   ```
   Sample Command: ./type_cast.out 3 9
   Sample Output: 27
   ```

5. (2 marks) Implement a basic calculator to evaluate a simple expression string. The expression string may contain open (and closing parentheses), the plus + or minus sign -, non-negative integers and empty spaces. *[Stack]*

   ```
   Sample Command: ./stack.out 10 - “(“ “(“ 2 + 8 “)” - 10 “)”
   Sample Output: 10
   ```

Deliverables:

1. ***Source code***: You need to provide the source code for the problem statements implemented using Rust. The source code should be in a .zip format and should be uploaded to moodle. A sample source code folder structure is shown below:

   ```
   Directory:  2020CSZ2445_Abhisek
               2020CSZ2445_Abhisek/map.rs
               2020CSZ2445_Abhisek/stack.rs
               2020CSZ2445_Abhisek/vector.rs
               2020CSZ2445_Abhisek/type_cast.rs
               2020CSZ2445_Abhisek/recursion.rs
   ```

   When we unzip the submission then we should see the above files in the aforementioned structure.

   The following procedure is for executing a Rust file.

   a. Your rust file should be compiled by the following command:

   ```
   rustc <filename>.rs -o <filename>.out
   ```

   b. The executable can be invoked by the following commands:

   ```
   ./<filename>.out <command line arguments separated by space>
   ```

Rubrics (10 marks)

For each of the problem statements, 1 mark for the basic test case and 1 mark for the advanced test case.

References

[1]: https://www.rust-lang.org/tools/install