

RedSet

RedSet creates a beneficial environment for learners that includes topic wise problems with solution, notes section where user can keep notes or templates, notify about running and upcoming contest of groups where user is joined, leaderboard and activity, and the exclusive feature LatticeLine, that includes compiler, groups feature (Individual can create or join groups. The teachers of every group can make announce, assignment - like Google Classroom and contest - like an Online Judge.)

Config

You can run the project in your environment through any IDE by cloning the project. Run `git clone https://github.com/Tamal267/RedSet` in your terminal. Before run, delete `RedSet/target` if found and create a database named `lattice` in your localhost mysql server. Run `CREATE DATABASE lattice;`. Then run the given sql script for the database `lattice`.

```
-- phpMyAdmin SQL Dump
-- version 5.1.1deb5ubuntu1
-- https://www.phpmyadmin.net/
--
-- Host: localhost:3306
-- Generation Time: Nov 20, 2023 at 01:35 PM
-- Server version: 8.0.35-0ubuntu0.22.04.1
-- PHP Version: 8.1.2-1ubuntu2.14

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `lattice`
--

--
-- -----
--
-- Table structure for table `announce`
--
```

```

CREATE TABLE `announce` (
  `text` text,
  `date` varchar(50) DEFAULT NULL,
  `gp` varchar(50) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

```

```

-- -----

```

```

--
-- Table structure for table `assignment`
--

```

```

CREATE TABLE `assignment` (
  `group_name` varchar(30) DEFAULT NULL,
  `text` text,
  `code` text,
  `input` longtext,
  `assignId` varchar(30) NOT NULL,
  `users` text,
  `timeLimit` varchar(5) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

```

```

-- -----

```

```

--
-- Table structure for table `conProb`
--

```

```

CREATE TABLE `conProb` (
  `problemid` varchar(30) NOT NULL,
  `statement` text,
  `code` text,
  `input` longtext,
  `users` text,
  `timeLimit` varchar(5) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

```

```

-- -----

```

```

--
-- Table structure for table `contest`
--

```

```

CREATE TABLE `contest` (
  `contestName` varchar(30) NOT NULL,

```

```

    `startTime` varchar(30) DEFAULT NULL,
    `duration` varchar(30) DEFAULT NULL,
    `problemsIds` text,
    `groupName` varchar(30) DEFAULT NULL,
    `ranking` text
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

```

```

-- -----

```

```

--
-- Table structure for table `gp`
--

```

```

CREATE TABLE `gp` (
  `name` varchar(30) NOT NULL,
  `stdents` varchar(1000) DEFAULT NULL,
  `teachers` varchar(1000) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

```

```

-- -----

```

```

--
-- Table structure for table `notes`
--

```

```

CREATE TABLE `notes` (
  `title` text,
  `note` longtext,
  `user` varchar(30) DEFAULT NULL,
  `date` varchar(30) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

```

```

-- -----

```

```

--
-- Table structure for table `problems`
--

```

```

CREATE TABLE `problems` (
  `problemid` varchar(50) NOT NULL,
  `statement` text,
  `code` text,
  `input` longtext,

```

```

        `users` text,
        `timeLimit` varchar(5) DEFAULT NULL
    ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

```

```

-----

```

```

--
-- Table structure for table `studyProblems`
--

```

```

CREATE TABLE `studyProblems` (
  `problemid` varchar(30) NOT NULL,
  `statement` text,
  `timelimit` varchar(5) DEFAULT NULL,
  `code` text,
  `input` longtext,
  `users` text,
  `solution` text
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

```

```

-----

```

```

--
-- Table structure for table `studyTopic`
--

```

```

CREATE TABLE `studyTopic` (
  `topicName` varchar(30) NOT NULL,
  `problemids` text
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

```

```

-----

```

```

--
-- Table structure for table `users`
--

```

```

CREATE TABLE `users` (
  `username` varchar(30) NOT NULL,
  `password` varchar(30) DEFAULT NULL,
  `connect` text,
  `fullName` varchar(30) DEFAULT NULL,
  `studentId` varchar(30) DEFAULT NULL,
  `email` varchar(30) DEFAULT NULL,
  `institute` varchar(30) DEFAULT NULL,

```

```

    `time` text
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

--
-- Indexes for dumped tables
--

--
-- Indexes for table `assignment`
--
ALTER TABLE `assignment`
  ADD UNIQUE KEY `assignId` (`assignId`);

--
-- Indexes for table `conProb`
--
ALTER TABLE `conProb`
  ADD PRIMARY KEY (`problemid`);

--
-- Indexes for table `contest`
--
ALTER TABLE `contest`
  ADD PRIMARY KEY (`contestName`);

--
-- Indexes for table `gp`
--
ALTER TABLE `gp`
  ADD PRIMARY KEY (`name`);

--
-- Indexes for table `problems`
--
ALTER TABLE `problems`
  ADD PRIMARY KEY (`problemid`);

--
-- Indexes for table `studyProblems`
--
ALTER TABLE `studyProblems`
  ADD PRIMARY KEY (`problemid`);

--
-- Indexes for table `studyTopic`
--

```

```
ALTER TABLE `studyTopic`
  ADD PRIMARY KEY (`topicName`);
```

```
--
-- Indexes for table `users`
--
```

```
ALTER TABLE `users`
  ADD PRIMARY KEY (`username`);
COMMIT;
```

```
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

Then change username and password in RedSet/src/main/java/com/example/RedSet/DBconnect.java.

Dashboard

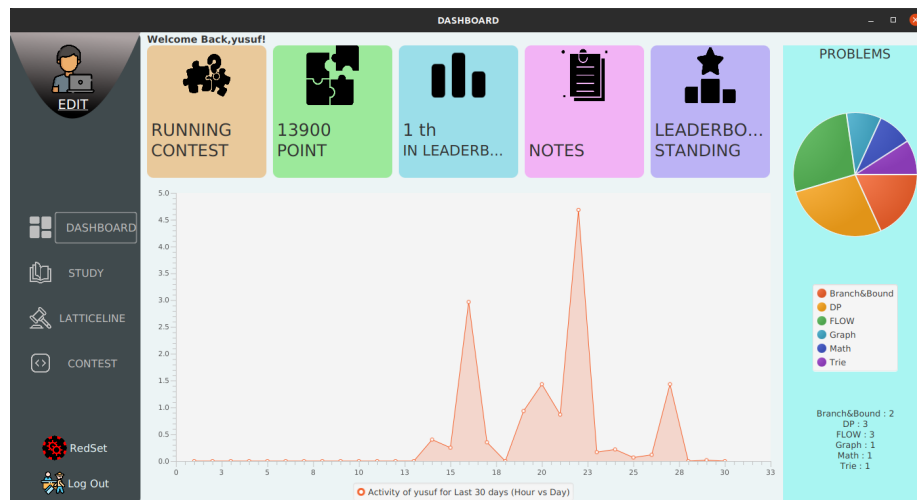
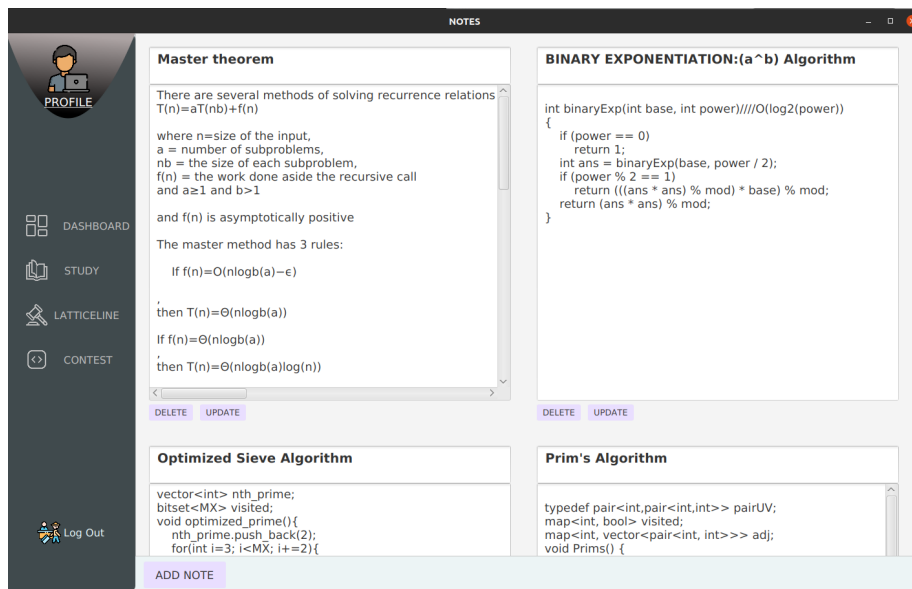


Figure 1: Dashboard

Dashboard gives a vivid idea of what can be done in RedSet. It gives important information to the users about how long he is in the application through graphical representation. Moreover, it also shows the number of different types of problems through a pie-chart. Besides it also provides features such as, * ### Notes:



It is possible to keep necessary codes, templates and notes inside this section.

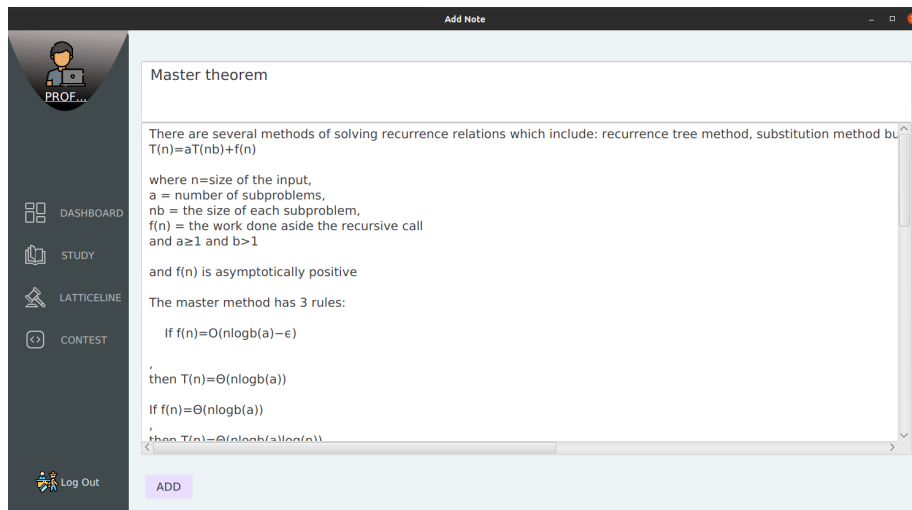


Figure 2: addnotes

A user can add/update and delete his desired template/code or notes.

-

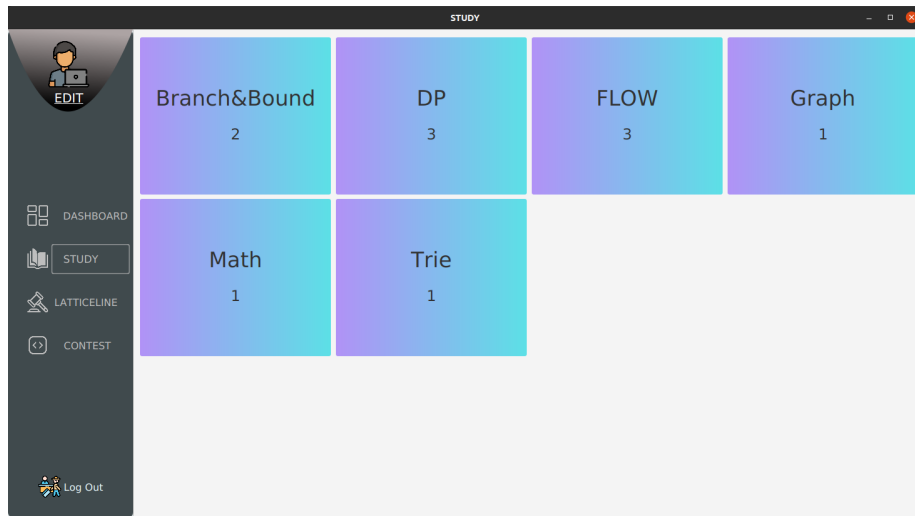


Figure 3: studyproblems

Study:

This section gives the user a chance to challenge his current capability by solving various types of problems.

contains problems of a certain topic.

User can attempt a problem and try solving it. If the solution provided by the user is correct, the judge will mark it accepted including required time.

User can read editorials or get the sample accepted solution if he fails to solve the problem.

-

Leaderboard:

Shows the user his current standing on the basis of the amount of time spent in the application.


-

Latticeline:

An exclusive part of RedSet which is discussed below:

Features of Latticeline

- Solving Problem from problem section
- Create Group, Join Group



Serial	Problem Name	Solve	Solution	Accepted
1	COIN CHANGE	Solve	Solution	1
2	An Easy LCS	Solve	Solution	1

DASHBOARD
 STUDY
 LATTICELINE
 CONTEST
 Log Out

BA...

Figure 4: problems of a particular topic

DASHBOARD
 STUDY
 LATTICELINE
 CONTEST
 LOG OUT

COIN CHANGE

Time Limit: 2s

Consider a money system consisting of n coins. Each coin has a positive integer value. Your task is to produce a sum of money using the available coins in such a way that the number of coins is minimal.
For example, if the coins are $\{1, 5, 7\}$ and the desired sum is 11, an optimal solution is $5 + 5 + 1$ which requires 3 coins.

Input
The first input line has two integers n and x : the number of coins and the desired sum of money.
The second line has n distinct integers c_1, c_1, \dots, c_n : the value of each coin.

Output
Print one integer: the minimum number of coins. If it is not possible to produce the desired sum, print -1.

Constraints
 $1 \leq n \leq 100$
 $1 \leq x \leq 10^6$
 $1 \leq c_i \leq 10^6$

Example
Input:
3 11
1 5 7

SOLUTION

```

37  return dp[k] = ans;
38  }
39
40  void solve()
41  {
42      int n, k;
43      cin >> n >> k;
44      vector<int> coins(n);
45      for (int i = 0; i < n; i++)
46          cin >> coins[i];
47      int ans = Number_of_coins(k, coins);
48      if (ans == inf)
49          cout << -1 << endl;
50      else
51          cout << ans << endl;
52  }
53
54  int32_t main()
55  {
56      YUSUF REZA HASNAT;
57      int t = 1;
58      // cin >> t;
59      while (t--)
60          solve();
61      return 0;
62  }

```

Choose File

Submit

2023/11/19 00:23:25
Accepted
Time: 20ms
Memory: 652KB

BACK

Figure 5: solve

DASHBOARD

STUDY

LATTICELINE

CONTEST

LOG OUT

EDITORIAL

EDITORIAL:

The coin change problem is a classic dynamic programming challenge that asks how many different ways you can make change for a specific amount using a set of coin denominations. Given a set of coin values and a target amount, the goal is to find the number of unique combinations to make change for that amount. Dynamic programming provides an efficient solution by building a DP table. Initialize the table with one way to make change for zero (no coins used).

Then, for each coin denomination and each possible amount from 1 to the target, calculate the number of ways by considering two cases: excluding the current coin (using the previous ways for the same amount) and including the current coin (using the ways for the remaining amount). Update the DP table with the sum of these cases.

The value in the bottom-right corner of the DP table represents the total number of ways to make change for the target amount.

This problem has practical applications in financial transactions and counting combinations.

SOLUTION

```

1 //!-----!//
2 //!          YUSUF REZA HASNAT          !//
3 //!-----!//
4
5 #include <bits/stdc++.h>
6 using namespace std;
7
8 #define int long long
9 #define float long double
10 #define pb push_back
11 #define v(v) (v).begin(), (v).end()
12 #define vr(v) (v).rbegin(), (v).rend()
13 #define endl "\n"
14 #define YUSUF ios::base::sync_with_stdio(false);
15 #define REZA cin.tie(NULL);
16 #define HASNAT cout.tie(NULL);
17
18 int mod = 1000000007;
19 float pi = acos(-1);
20 int inf = INT_MAX;
21 const int N = 2 * 1e6 + 6;
22 vector<int> dp(N, -1);
23
24 int Number_of_coins(int k, vector<int> &coins)
25 {
26     if (dp[k] != -1)
27         return dp[k];
28     if (k == 0)
29         return 0;
30     int ans = inf;
31     for (auto i : coins)
32     {
33         if (k - i >= 0)
34             ans = min(ans, Number_of_coins(k - i, coins) + 1);
35     }
36     dp[k] = ans;
37     return ans;
38 }
39
40 int main()
41 {
42     int n, m;
43     cin >> n >> m;
44     vector<int> coins;
45     for (int i = 0; i < n; i++)
46     {
47         int x;
48         cin >> x;
49         coins.push_back(x);
50     }
51     int ans = Number_of_coins(m, coins);
52     cout << ans << endl;
53     return 0;
54 }

```

BACK

Figure 6: editorial

DASHBOARD

STUDY

LATTICELINE

CONTEST

Log Out

RANKING

RANK	USERNAME
1	yusuf
2	tamal_63
3	tamal63

Figure 7: leaderboard

- Teachers of a group can make assignment, announce & contest
- Assignment of a student will accept when all the testcase for this assignemnt passed, like an online judge
- Teachers can see the status of every assignment
- Realtime ranking of a contest
- Others solution of a contest will visible after the contest
- Contest and Assignment is editable/updatable.
- Teachers can add a user as teacher from the group
- And more

Problems Section

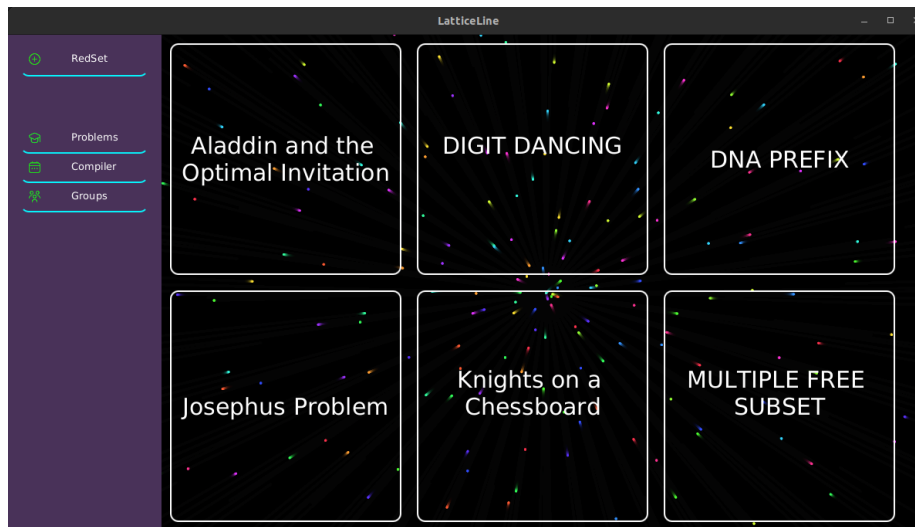


Figure 8: problem_section

It is global problem section. Any user can solve problem here.

Compiler Section

It is a simple editor with highlight C++ code feature that shows output including time and memory complexity. You can choose a file to compile from choose button. Only C++ is available right now.

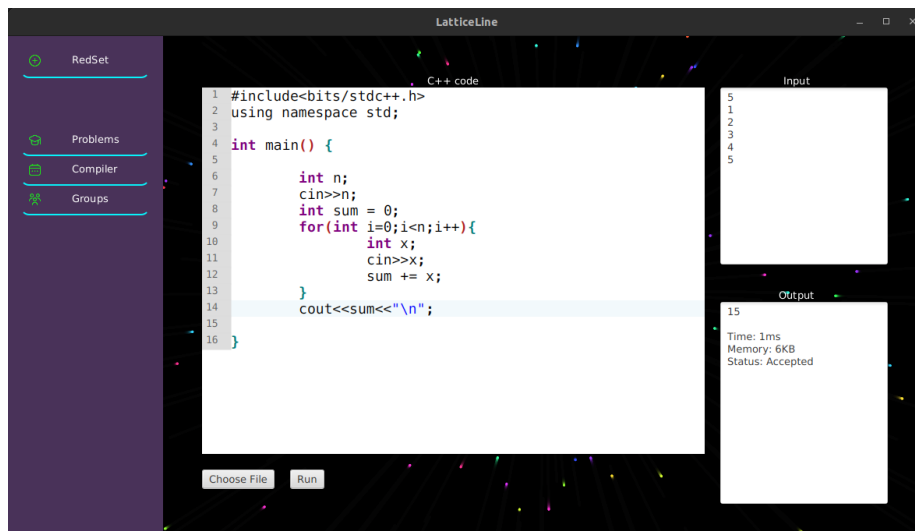
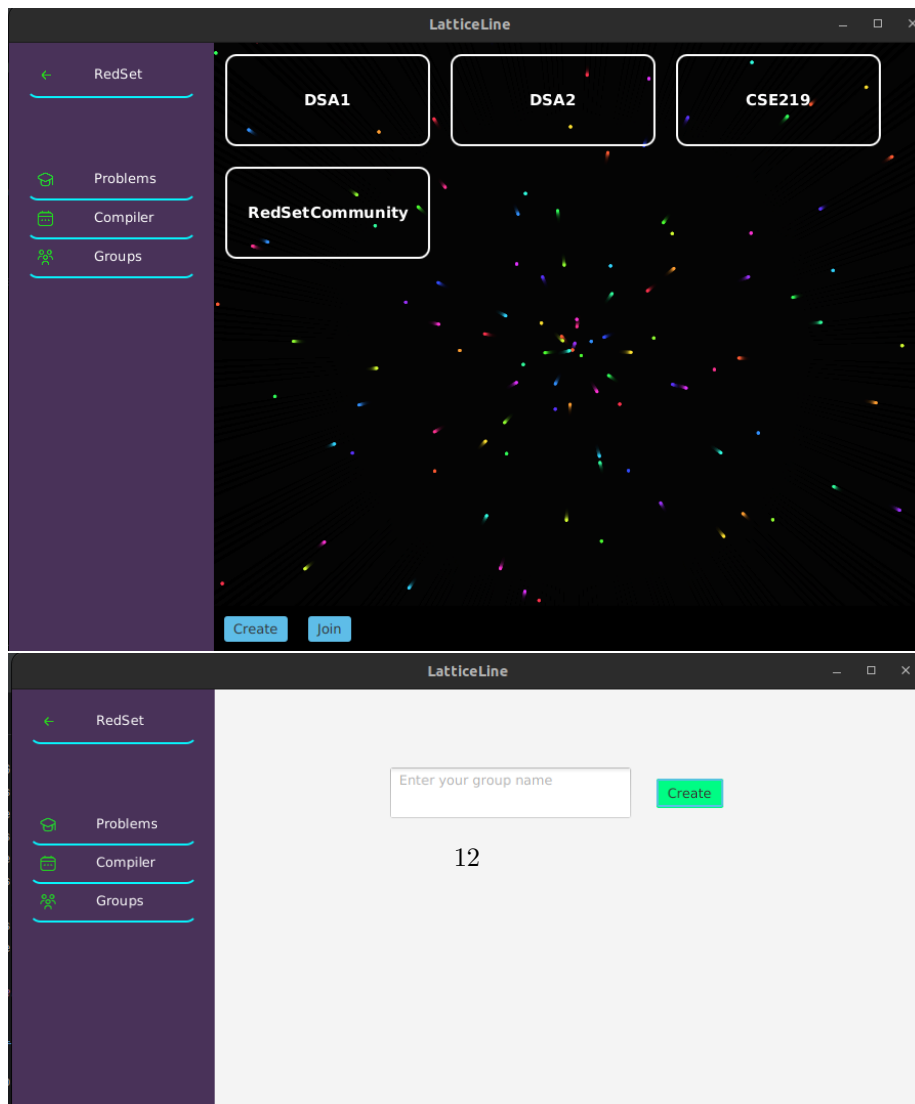
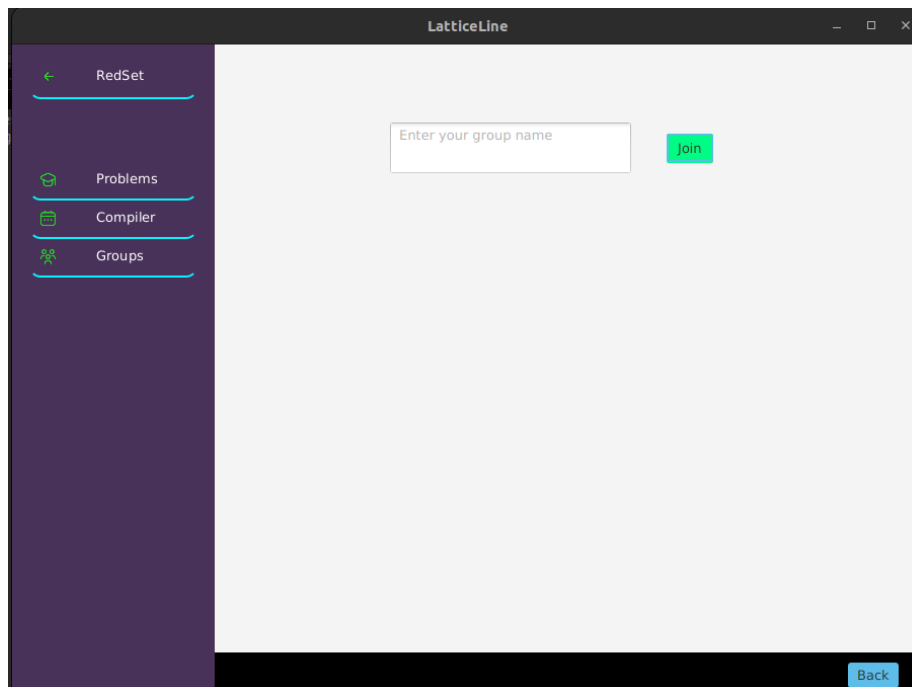


Figure 9: compiler_section

Groups Section

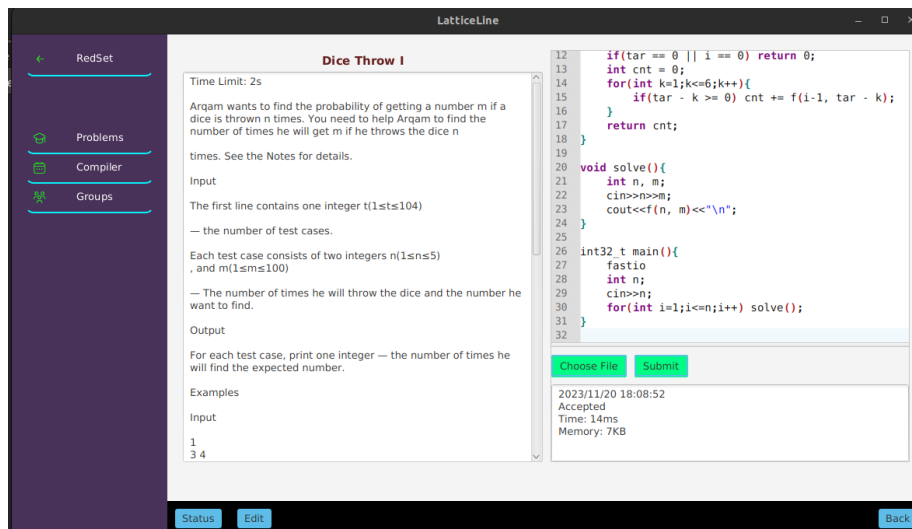




You can create a group. Group name will unique. You can join a group as a student also.

Create Assignment, like Google Classroom. Assignment are showing in each of the boxes.

Only Teachers can create assignment.



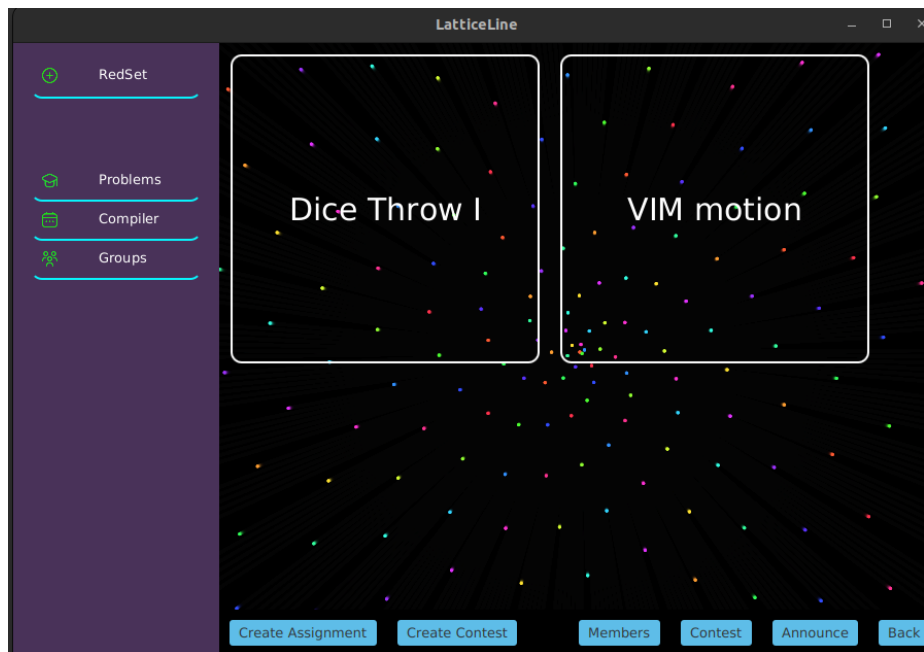
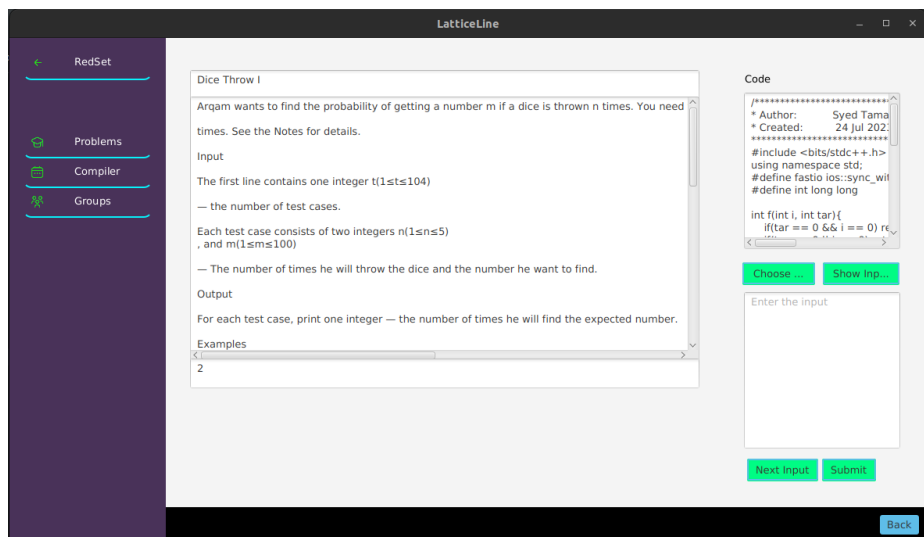
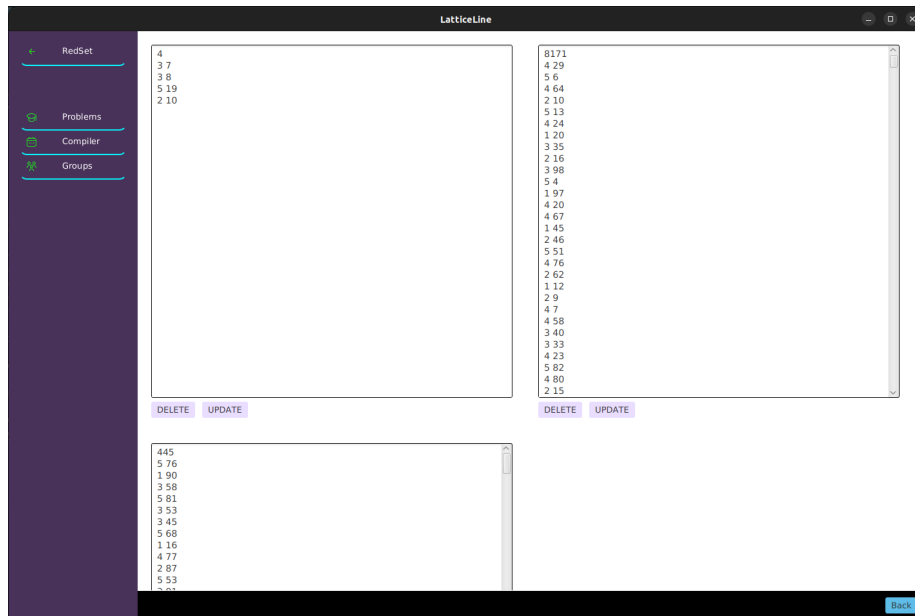


Figure 10: eachgroup_section



The screenshot shows a web application window titled "LatticeLine". On the left is a dark purple sidebar with four menu items: "RedSet" (with a green plus icon), "Problems" (with a green book icon), "Compiler" (with a green code icon), and "Groups" (with a green group icon). The main content area is light gray and contains several input fields and buttons. On the left side of the main area, there are three stacked text input fields: "Enter your assignment name", "Enter the problem statement" (with a larger text area below it), and "Enter time limit in second". On the right side, under the heading "Code", there is a text input field "Enter the accepted code", a green "Choose File" button, another text input field "Enter the input", and two green buttons labeled "Next input" and "Submit". At the bottom right of the main area is a blue "Back" button.

Figure 11: crtassign_section



An assignment. Previous accepted solution remains visible with time & space complexity if any. Only teachers can show status and edit the Assignment. Teachers can see inputs and update inputs.

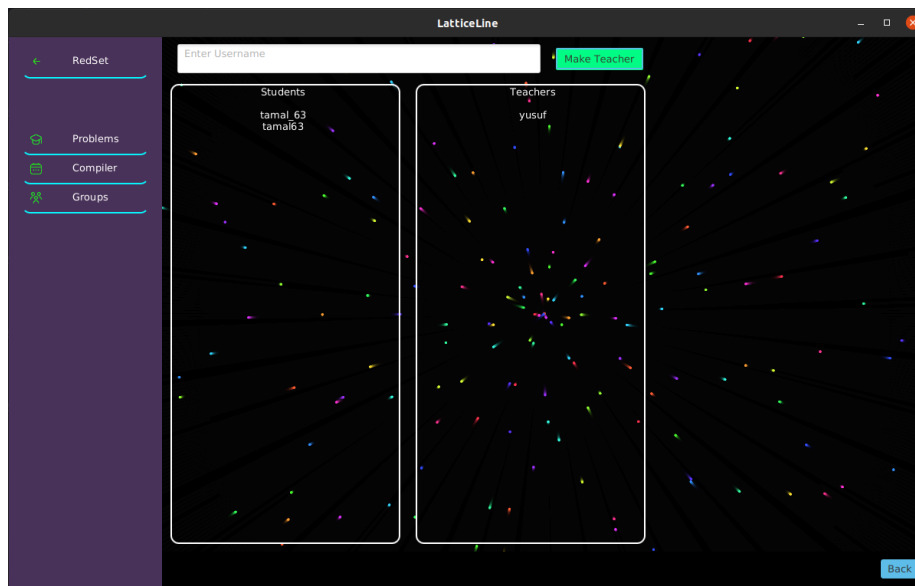


Figure 12: maketeacher

A teacher can promote a student as teacher.

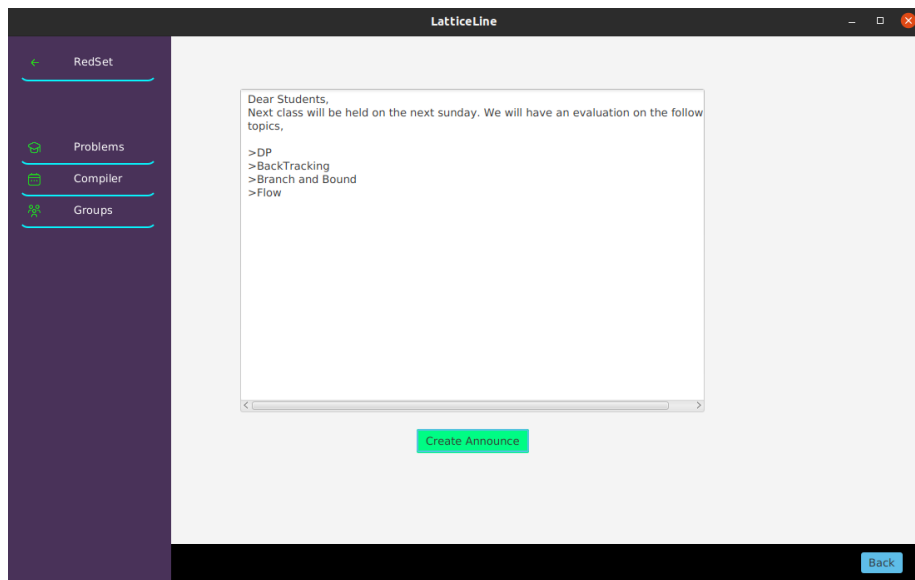


Figure 13: createannounce

Teachers can make necessary announcements if required!

Users can view the current announcements that were made up until the teacher doesn't delete it.

Students can enter to the contest if the contest is in running or in ended state. Teachers can edit/update contest anytime.

You can see a countdown at the bottom inside a contest page.

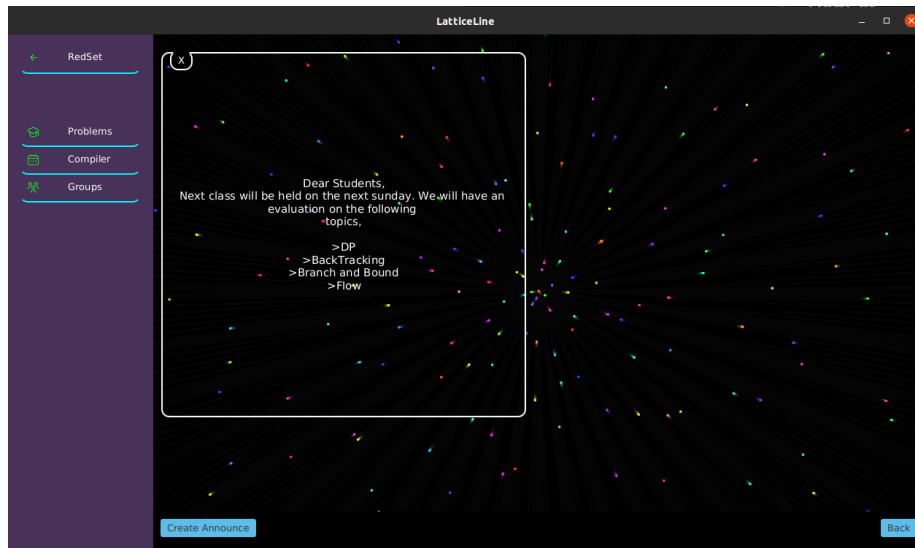


Figure 14: viewannounce

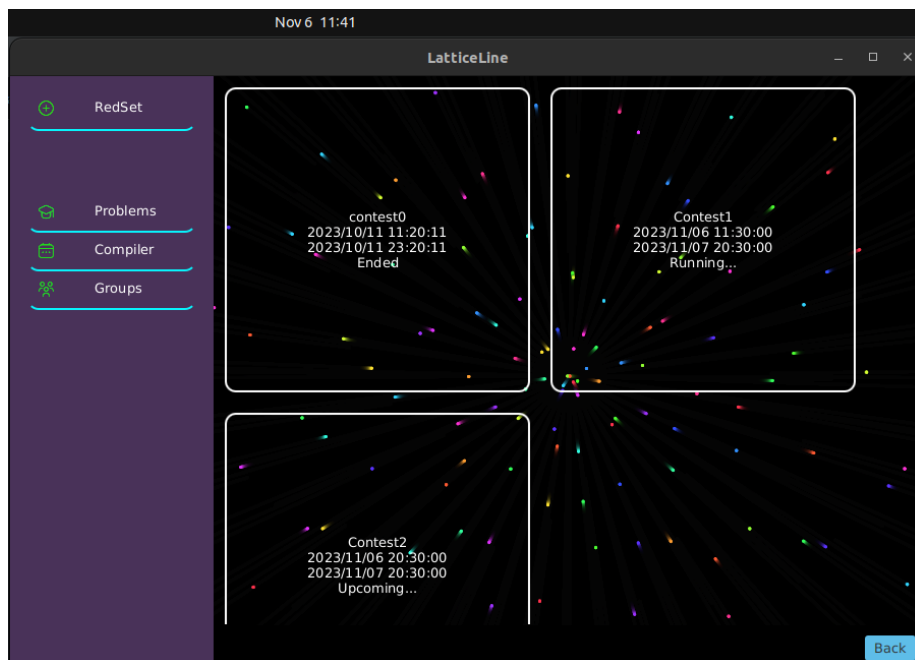


Figure 15: contest_section

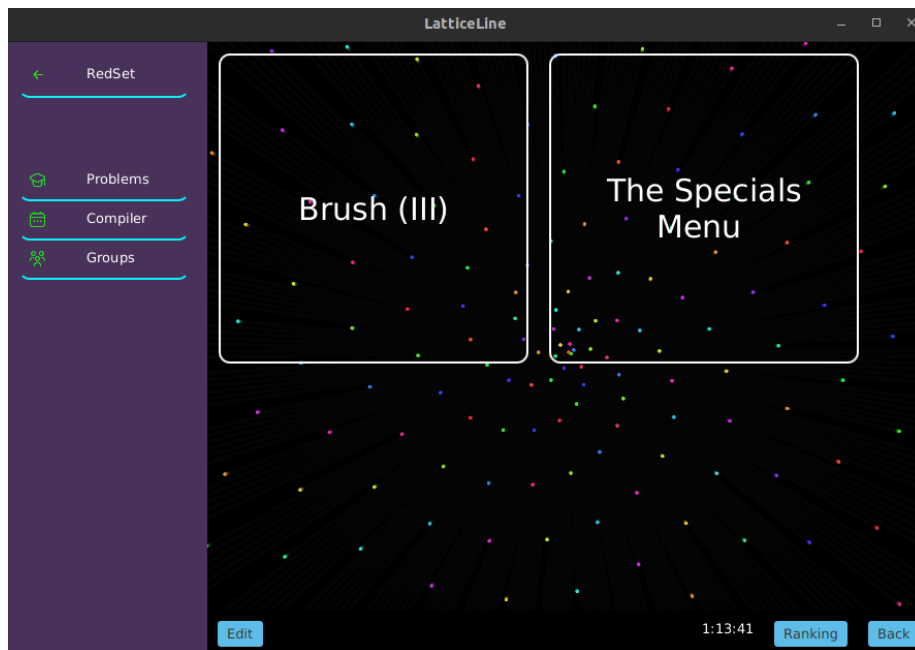
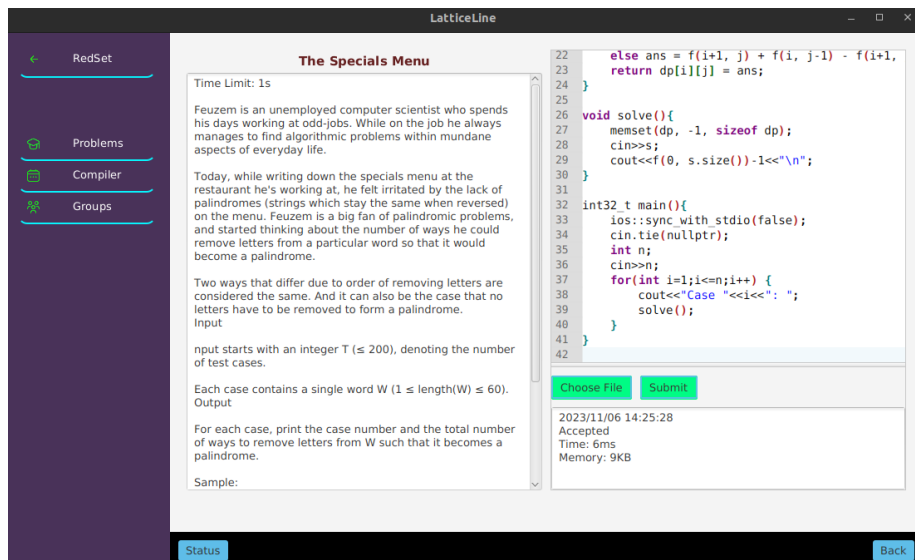
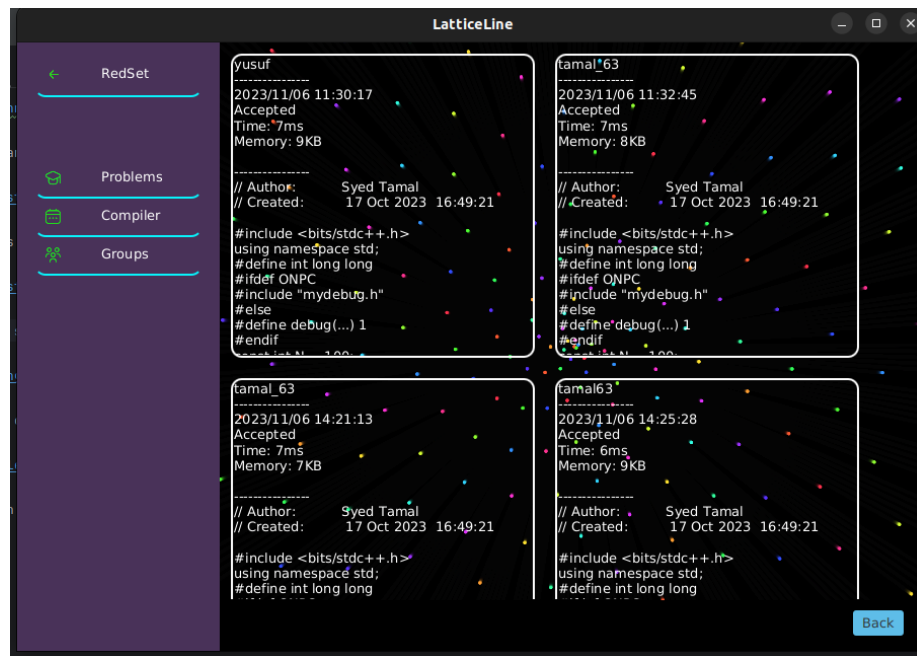


Figure 16: contest_section





Students can only see the status after the end time of a contest.

Ranking of every contest. 20 penalty increase for a wrong submission.

User can view his current given information and edit them anytime.

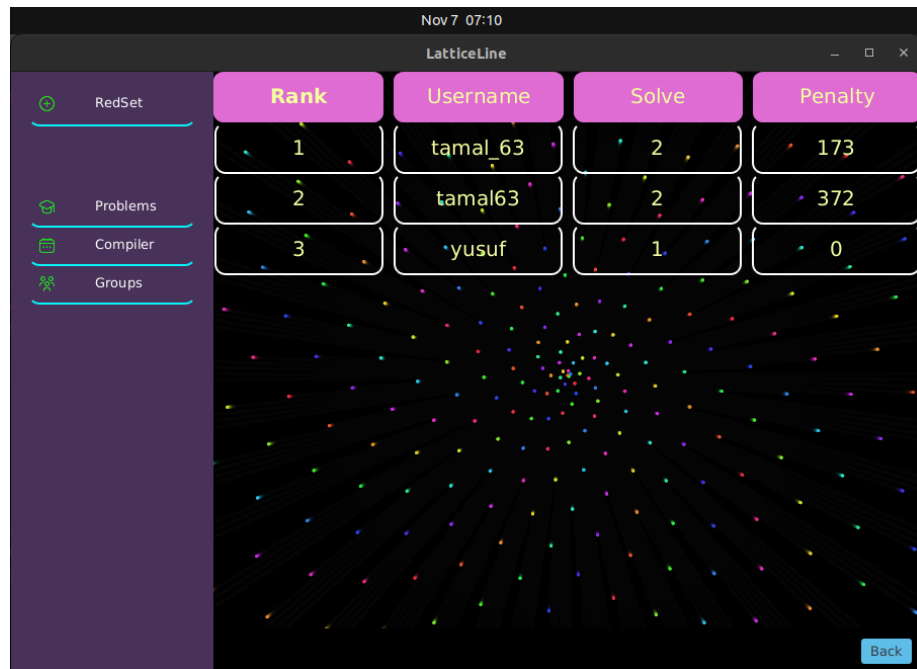



Figure 17: ranking_section

PROFILE



FULL NAME
Yusuf Reza Hasnat

USERNAME
Hasnat0006

STUDENT ID
202214112

EMAIL
yusufrezahasnat0006@gmail.com

OLD PASSWORD
Old password

NEW PASSWORD
New password

RE-TYPE NEW PASSWORD
Re-type new password

BACK **SAVE**

Figure 18: profile