# Programming assignment:

# A video game with a database-backed leaderboard

|  |  |
| --- | --- |
| Finally, we combine the queries from Task 1 into the Java code from Task 3. Every time a user plays the game, when the game is over, after the new record is added to the database, the program needs to display the high scores and the leaderboards so the player can see how they stack up to the competition. This is the goal of Task 4. | Space Invaders screenshot |

**INVADERS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **id** | **player** | **hero** | **gametime** | **score** |
| 1 | crono | 1 | 2017-11-14 12:17:01 | 50 |
| 2 | crono | 2 | 2017-11-14 12:17:01 | 470 |
| 3 | marle | 3 | 2017-11-14 12:17:01 | 320 |
| 4 | lucca | 2 | 2017-11-14 12:17:01 | 5630 |
| 5 | crono | 1 | 2017-11-14 12:17:01 | 480 |

## Task 4: Display the leaderboards

1. Revisit Task 1. Look for the corrections the instructor put as annotations on your submission on Canvas. Figure out the correct queries.
2. Open the NetBeans project that you worked on in Task 3. Find the point in the code after the new record is inserted into the database, but before the database connection is closed.
3. In Task 1, Problem 1, we wrote a query to find Crono’s high score. Adapt this so that the player name is a parameter in a prepared statement. Use the player’s screenname to set the parameter. The Java code for this is almost exactly the same as looking up the manager by ID in the Lab. I will leave this one to you. Print out the high score so the player can compare their new score to the high score.
4. In Task 1, Problem 2, we wrote a query to find the list of high scores. The problem didn’t say, but let’s limit it to the top 20. I’ll walk you through this one, so you can complete the others on your own.
   1. We need a prepared statement containing the SQL query. We don’t need any parameters for this or the rest of the problems.

PreparedStatement q2 = conn.prepareStatement(

"SELECT score, player, hero FROM invaders " +

"ORDER BY score DESC LIMIT 20");

* 1. Execute the query:

ResultSet r2 = q2.executeQuery();

* 1. We need to print a table. We start with the header of the table:

System.out.println("---------------------------------");

System.out.println(" Top 20 High Scores");

System.out.println("---------------------------------");

* 1. Next we loop over the record set and get the values out of it:

while (r2.next())

{

int hscore = r2.getInt("score");

String hplayer = r2.getString("player");

int hhero = r2.getInt("hero");

* 1. Now we need to print a row of the table. Here we use a Java method called printf. It lets you print data in a nicely formatted way. You can look this up for details, but put simply, %d is for an integer, %s is for a string, and %n prints a new line. You can also specify how wide you want the column, and it fills in the rest with spaces. Here’s what we have:

System.out.printf("%9d %-16s %-6s",

hscore, hplayer, heroNames[hhero-1]);

System.out.println();

It’s width 9 for the score, 16 (left aligned) for the screenname, and 6 (left aligned) for the hero name.

* 1. Finally, finish the loop and print a bottom for the table.

}

System.out.println("---------------------------------");

* 1. Try it out.

1. Task 1, Problem 3, was a leaderboard for each hero. Write code similar to the above to print 3 more tables—one for Currer, one for Acton, and one for Ellis. Each query should have a GROUP BY player clause, because we want to see each player only once (or possibly not at all) in each leaderboard.
2. Task 1, Problem 4, was an overall leaderboard for the game. This one was more complex because I wanted to see the details of the game in which the high score was achieved. Print a table for this one, too. Here is a query you can use:

SELECT U.player, U.hero, U.gametime, U.score

FROM invaders U, (

SELECT MIN(id) AS first\_id

FROM invaders S, (

SELECT player, MAX(score) AS high\_score

FROM invaders

GROUP BY player

ORDER BY high\_score DESC

LIMIT 10

) H

WHERE S.player = H.player AND S.score = H.high\_score

GROUP BY S.player

) F

WHERE U.id = F.first\_id

ORDER BY U.score DESC

1. Play the game several times, and give it to your friends to play. Check the leaderboards to see if they look right.
2. Upload the ScoreUpdater.java file to Canvas by the posted due date.