**CENG 342 hw – 1** (Due 02/04/2021)

**Subject:** Playing a rock paper scissors game with MPI.

**Aim:** The aim of this project is to use MPI processes and gain some experience about distributed memory parallel programming.

|  |
| --- |
| Cheating and studying together with the other students is strictly prohibited. A student should never show his/her codes to another student whether in seeking aid or giving it. A student may ask a fellow student general questions which do not pertain to a particular programming assignment but may not ask any programming aid.  Disciplinary action to be taken against cheating is arranged by the Rules and Regulations Governing Student Disciplinary Actions in Institutions of Higher Education. |

**Algorithm:**

**There will be 2 MPI process**

1. Process 0 will be responsible from printing all the outputs
2. 2 Processes will guess an item randomly in parallel
3. Then they will compare their items
   1. Process-1 should send its item to Process-0, that also means, Process-0 should receive that item to Process-1
   2. Process-0 compare the items and acknowledge the Process-1 about the winning process
4. Score of the winning process will be increased
5. Game will continue until one of the process gains 5 points.

The game has only two possible outcomes: a draw, or a win for one player and a loss for the other. A player who decides to play rock will beat another player who has chosen scissors, but will lose to one who has played paper; a play of paper will lose to a play of scissors. If both players choose the same shape, the game is tied and is usually immediately replayed to break the tie.

For inter-process communications you should use MPI.

**Requirements:**

1. An at least 1 page report. The format of the report should be the same with hw1\_XX\_report.pdf. Replace XX with your group student ID.
   1. The report should contain a summary of your homework, implementation details, how you used MPI in homework, what is the most changeling part.
   2. The screenshots of outputs of your program after running your program 3 times successively. The scores of each game should be different since they are based on totally random decisions. Take necessary actions to produce random decisions for every game. That is, ensure that the score for two consecutive games will always be different.
   3. **Convert it to a PDF file. (otherwise -20 points)**
2. Writing and reading directly from a file is strictly prohibited. (gets 0 point)
3. You will use C programming language in Linux environment. Your program consists of a single file called **game.c** . Your program will not take any input. It should be compiled with **mpicc -o game game.c** and run with **mpirun -n 2 game .**
4. You will upload your homework to aybuzem.aybu.edu.tr after compressing them in a single ZIP file name **hw1\_XX.zip,** replace XX with your group student ID. The compressed ZIP file should include:
   1. game.c C source file.
   2. a file which includes compilation commands that you used (mpicc … -o ….. etc.), so it is a text file that will contain just one line.
   3. A report in PDF format.
5. The codes that give compilation error or run-time error will get **0.**

**Example:**

Here is some example executions for the homework.

**>> mpicc game.c -o game**

**>> mpirun -n 2 ./game**

The game starts

--

Turn 1, Process-0: ROCK, Process-1: ROCK

Draw, Score: 0 – 0

–

Turn 2, Process-0: SCISSORS, Process-1: ROCK

Parent win, Score: 1 – 0

--

Turn 3, Process-0: ROCK, Process-1: PAPER

Child win, Score: 1 – 1

–

Turn 4, Process-0: PAPER, Process-1: SCISSORS

Child win, Score: 1 – 2

–

Turn 5, Process-0: ROCK, Process-1: ROCK

Draw, Score: 1 – 2

–

…

…

…

--

Turn 17, Process-0: PAPER, Process-1: ROCK

Process-0 win, Score: 5 – 4

–-

Process-0 has won the game with score: 5 – 4 in 17 Turns.

–-

The game ends

Please do not try to put some fake results in your report. TA and I will test, run and analyze your projects separately on our computers.

Good luck!