

CSE102 – Computer Programming with C (Spring 2019)

Summer Project – Boulder Dash

Handed out: 9:00am Thursday June 27, 2019.

Due: 23:55pm Friday September 13, 2019.

Hand-in Policy: Hand in via Moodle. No late submissions will be accepted.

Collaboration Policy: No collaboration is permitted.

Grading: This project will contribute some points towards your grade for CSE241 in Fall 2019 (exact contribution is up to the CSE241 course instructor).

Description: You will implement a version of the game Boulder Dash in C programming language using SDL.

SDL: SLD (<http://www.libsdl.org/>) library is a cross-platform library for developing graphics-based application. Use this library to develop your C program implementing the game described below.

Boulder Dash: This is a vintage maze puzzler in which players collect diamonds before time runs out. A few screen-shots from some original games are provided below. You can also see the video (https://www.youtube.com/watch?v=FiEVfa1OK_o) for the C64 version of the game.



- Music:
 - A background music should always be playing while the game is in session.
 - This music should increase in rhythm within the last 30%, 20% and 10% of the allowed time for the current level by 10%, 30% and 60% respectively.
- Cells and items: The game board is composed of square cells in grid formation. The total size of the grid defines the map of the level. A cell can be empty or can include any of the following items. Some of the cells are shown in the following illustration.



- Empty cell – Cells that do not have any items in them. The miner, spiders and monsters can move freely into these cells.
- Miner – This is the only directly controllable item by the user. The miner can move in any direction if the cell in the direction of the movement:
 - is empty, or
 - has a diamond in it (in which case the miner takes the diamond with the attached score and occupies the cell), or
 - has a rock and the next cell after the rock in the direction of the movement is empty (in which case the miner and the rock moves one cell towards the direction of movement), or
 - has a spider or monster in it (in which case the miner is destroyed losing session), or
 - has a door (in which case the game goes to the next level).

The miner should not ever be in the same cell as the spider or monster. In either case, the miner will be destroyed by the

- Earth – A miner can occupy earth cell at any time. The cell then becomes empty. Spiders and monsters cannot occupy the cells with earth in them. If a spider or monster is destroyed, the adjacent cells with earth may be converted to diamonds.
 - Border – a fixed cell that cannot be moved or destroyed.
 - Rock– a cell with a rock does not disappear but the rock can be moved left or right if the adjacent cell in the direction of movement is empty. The rock can free fall when there is nothing underneath. If in free fall and lands on a spider or monster, rock destroys them turning the adjacent earth cells to diamonds. Once moved, the original cell becomes empty.
 - Diamond – a collectable item by the miner. Once collected, the cell becomes empty.
 - Spider – an item that moves always to the next available space (in the following order – right, bottom, left or top). If the miner is in the adjacent cell (four-neighborhood), the miner is killed. If a solid rock lands on a spider, all the adjacent 8 earth cells turn into diamonds.
 - Monster – like spiders but tries to catch the miner by the calculating the shortest path (Manhattan distance) at any given time. If a solid rock lands on it, the surrounding 12 cells including earth turn into diamonds.
 - Water – fills the surrounding four (top, bottom, left and right) earth cells for every N seconds. N is a fixed number and defined by the level. The miner cannot occupy or move the cells with water.
 - Door – when reached, the current level can be completed and the player can move to the next level. Exit door appears only after the required number of diamonds are collected.
- You should design a game with 10 levels.
 - The user should have a maximum number of lives. The number of lives can be updated after a certain number of levels achieved.
 - Each level should have a maximum allowed time. After which the user will lose a live and start the level from scratch.
 - A few levels can be just a single screen.

- At least 4 levels should cover multiple screens.
 - The actual mine should be at least four screens wide and high.
 - The miner should always be in the center of the current screen unless on close to the boundary in which case the mine boundary and screen boundary should be the same.
- GUI
 - The screen size should be adjustable. Cell size should be fixed (no smaller than 30 pixel wide or high).
 - Game level and scores should be shown to the user.
 - Keyboard arrows should be used for movement of the miner (left, right, top and bottom).
 - A pause button should allow the user to pause the game while a resume button should allow the game to resume.

What to hand in: You are expected to hand in all your source code (library and main program) along with your makefile in a ZIP or similarly archived file named **"CSE102_SP_LastName_FirstName_StudentNo.zip"**. When the makefile is run, it should compile everything and produce an executable that runs the game. The target platform will be Ubuntu 14 or higher. The submitted folder should include all the third-party libraries (anything not included in a standard distribution of Ubuntu and gcc). The executable program should illustrate all the above functionality.