Homework 3

Computer Archiecture and Organization

Sofware Engineering Program,

School of Computer Engineering, KMITL

67011352 Theepakorn Phayonrat

Hurkle Game

How to compile and Run

```
as hurkle.s -o hurkle.o
ld hurkle.0 -o hurkle
./hurkle
```

Design and Implementation Details

Structure

1. `init_game`

   - **Purpose**: Sets up the initial game state.
   - **Description**: Calls `init_random` to seed PRNG, then generates two random coordinates using `generate_one`$_r andom\_number$.

2. `init_random`

   - **Purpose**: Initialize seed for random.
   - **Description**: Opens `/dev/urandom`, get seed and initializes `PRNG`.

3. `generate`$_o ne_r andom_n umber$

   4. **Purpose**: Generate a random number.

   5. **Description**: Generates a random number and stores into `r0`.

`game_loop`

Purpose: Control the main flow of the game.

Description:

1. Check whether player guesses 10 time.
   - **Yes**: End game with player lose.
   - **No**: Increment the guess count and continue to get player input.
2. Check whether player guess is correct.
   - **Yes**: End game with player won.
   - **No**: Give hint then continue looping until game end.

`get_player_guess`

- **Purpose**: Take player input to continue the game.
- **Description**: Takes player input and convert to 2 integers to check the hurkle location.

`give_feedback`

- **Purpose**: Help player find hurkle
- **Description**: Compares the X Y coordinates to give hint to player
  - Compares Y coordinates: prints "Too high!" or "Too low!"
  - Compares X coordinates: prints "Too far left!" or "Too far right!"
  - Calculates Manhattan distance: $|GX - HX| + |GY - HY|$.
    If Manhattan distance $\leq 2, prints$`"You are very close!"`

`print_string`

- **Purpose**: Print string as ASCII output
- **Description**: Uses `SVC` with 4 in `r7`.

`itoa`

- **Purpose**: Convert integer to ASCII.
- **Description**: Builds string backward and copies to buffer by repeatedly dividing that integer by 10.

Key Designs Choices

1. **Randomness**: Use `/dev/urandom` to random number for this program.
2. **Parsing integer to ASCII from user input**: Since the game need only 2 of 1 digit number from 0 to 9, we can split by skipping whitespace, take those 2 input and subtract by `'0'` to get integer value of the input.

Challenges faced with Solutions

Challenge	**Solution**
ARM System Calls are complicate to understand if you do not have that much knowledge in this field	Go to here and read the specifications.
We need to convert integer back to ASCII every time we output to the terminal.	Create `itoa` to convert back to ASCII.

Game Demo

https://youtu.be/1oMMeVF0yOg