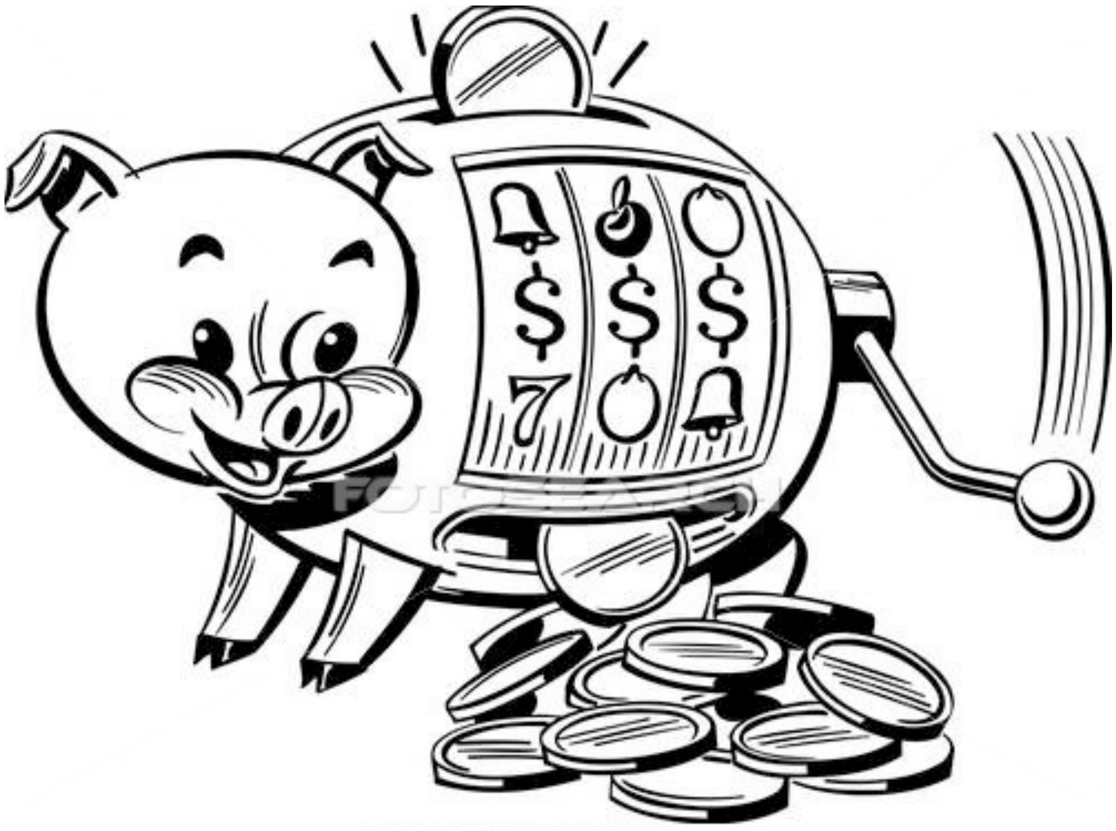
Software Engineering Project 1 (COMP 10050)

Assignment 1 – Implementation of a Simple Slot Machine

Aim of the assignment: *to create a simple command line version of a slot machine.*



A. Detailed Specification

For this assignment, you are asked to implement a command line version of a slot machine in C. The slot machine consists of three columns, and each column can hold one of three possible values: 'APPLE', 'ORANGE' or 'PEAR'.

When the slot machine begins, the player is asked to bet X amount of their initial credit, **I** (default value for **I** should be 10 credits), where **X ≦ I**. For each play of the slot machine there are three possible outcomes:

1. **Full house** – all 3 columns contain the *same* value. Player wins the amount they bet (i.e. X). X (winnings) should be added to current credit balance (or I if initial run) to form new credit balance.
2. **Half house** –2 of the 3 columns contain the *same* value. Player wins half the amount they bet (i.e. X/2). X/2 (winnings) should be added to current credit balance (or I if initial run) to form new credit balance.
3. **Empty house** - all 3 columns contain *different* values. Player **loses** X credits. X credits should be removed from current credit balance (or I if initial run) to form new credit balance.

After the first play, the user should be offered the choice of playing again by entering 'y'/'n'. The game then proceeds as follows:

- If 'y' is entered and the user has credit then they are allowed play again and proceed by entering their new bet amount X.
- If 'y' is entered and the user has NO CREDIT, then the game terminates by displaying a message such as *"Sorry, but you ain't got no credit - Bye!!"*
- If 'n' is entered and the user has made a loss overall then a message similar to "`******** End of Game: total amount LOST = -10 credits *******`"
- If 'n' is entered and the user has gained credits overall, a message similar to "`******** End of Game: total amount WON = 30 credits ********`"

Additional Notes

- Each bet amount X should be greater than 2 and ≦ available credit. If X is not valid, a message similar to: "Invalid bet (must be greater than 2 credits and ≦ to available credit), try again:" should be displayed.
- Checks to ensure that the user gave the correct input should be included. For example, when asking user for bet amount, you should have code in place that the value entered was a int and not a char[]. If incorrect a message asking the user to enter the value again should be displayed.

To make all this clearer, I have included some sample runs of the system in Section D below.

B. Code Design Requirements

- **Use Structs.** You must use *structs* to represent a *Column* and an individual *Slot*. **Hint:** a Slot consists of three Columns
- **Create functions where appropriate.** See lecture 2 for examples

C. Your Submission

1. Create a new project.

In Eclipse, create a new C project called "*SlotMachine*". Within this project, create a c source file called "SlotMachine.c" to hold your solution to this assignment.

2. Document your code.

You must comment your solution as follows:

1. You should include a short comment at the start of your main c source file which describes generally how the code works (e.g. describe inputs for the game etc).
2. For each function, you should describe (in a few sentences) the purpose of the function, any parameters of the function and possible return types the function may have.

3. Submitting your solution.

Once you've made your final changes to your "*SlotMachine*" project and made sure your code compiles and runs correctly, you should submit the source file through the

Moodle page. You should also submit a text file containing sample runs of the program.

D. Sample Runs of the Game

Here are some examples of sample runs of the game, designed to illustrate the game’s functionality:

```
----------------------------- Sample Run 1 -----------------------------------------------
******** Welcome to my Slot Machine ********
Your available credit is 10 (10 is the default starting credit)
How much you want to bet? (user entered 10)

Your Selection: |APPLE| |PEAR| |ORANGE|

*** You lost 10 credits ***

Current Credit Balance = 0
Play again? ('y/n'): (user entered ‘y’)
Sorry, but you ain't got no credit - Bye!! (if you have no credit, the program should terminate
with this message if user entered ‘y’ – see below for case where user enters ‘n’)
------------------------------------------------------------------------------------------
```

```
----------------------------- Sample Run 2 -----------------------------------------------
******** Welcome to my Slot Machine ********
Your available credit is 10
How much you want to bet? (user entered ‘10’)

Your Selection: |ORANGE| |APPLE| |PEAR|

*** You lost 10 credits ***

Current Credit Balance = 0
Play again? ('y/n'): (user entered ‘n’)
******** End of Game: total amount LOST = -10 credits ******** (if you have no
credit, the program should terminate with a similar message  (reporting how much the user
lost ), if user entered ‘n’)
```

```
----------------------------- Sample Run 3 -----------------------------------------------
******** Welcome to my Slot Machine ********
Your available credit is 10
How much you want to bet? (user entered 20)
Invalid bet (must be greater than 2 credits and <= to available credit), try again:
(user entered 10)

Your Selection: |ORANGE| |ORANGE| |ORANGE|

*** All 3 the same - You won 10 credits ***
```

```
Current Credit Balance = 20
Play again? ('y/n'): (user entered ‘y’)

Your available credit is 20
How much you want to bet? (user entered 20)

Your Selection: |ORANGE| |APPLE| |APPLE|

*** Half House - You won 10 credits ***

Current Credit Balance = 30
Play again? ('y/n'): (user entered 50 by accident)
Incorrect Input (must be 'y/n') - Play again? ('y/n'): (user entered ‘n’)
******** End of Game: total amount WON = 30 credits ********
--------------------------------------------------------------------------------------------
```