CMP-104-FruitMachine

Features:

- Cross compatability with both windows and *nix (tested with linux)
- Written to use the ncurses library on *nix and a pre-built version of PDCurses on Windows
- cmake based build system for linux
- fancy build shell script for linux

Dependencies

Windows

- Visual Studio 2019

Linux

- cmake (>=3.15)
- C++ 14 Compatable compiler (i.e GCC5 or newer, tested with GCC 9.2.0)
- Make
- ncurses

Ubuntu 16.04

```
sudo apt install build-essential cmake libncurses5-dev
```

Arch linux

```
sudo pacman -S base-devel cmake ncurses
```

Building from source

Windows

Option 1:

1. Import Visual Studio Project from src/VS2019
2. Build

Option 2:

1. Install Ubuntu via Windows Subsystem for Linux
2. Follow Linux Install Instructions

Linux

1. Install Dependancies
2. `sh run.sh`

Design decisions

- As all my PCs are linux based I needed to write this project in such a way that it would work on linux
  - i.e without `Windows.h` or `conio.h`
- My first idea was to use no libraries and just use different code depending on which OS it was built on
  - see src/Take1.cpp
  - this used preprocessor directives to select diffent parts of the code depending on OS
  - it was scraped due to there being no common way of easily getting input
- This lead me to search for a windows compatible curses library - PDCurses
- The next point was a cross compatable build system - cmake
  - this was easy on linux as I can simply link to the system ncurses libraries
  - no so on windows, I needed to build those my self
  - PDCurses provides makefiles for various targets including both UNIX and MS Visual Studio
    - But no CMakeLists.txt. 😦
  - In the end I ended up using two build systems
    - cmake for *nix
    - MSBuild for Windows

Future development

- Universal cmake build system
- Fancy things like:
  - A betting system
  - Accounts
  - Unicode

Know Issues

- When displaying characters, the characters address is printed rather than the character its self