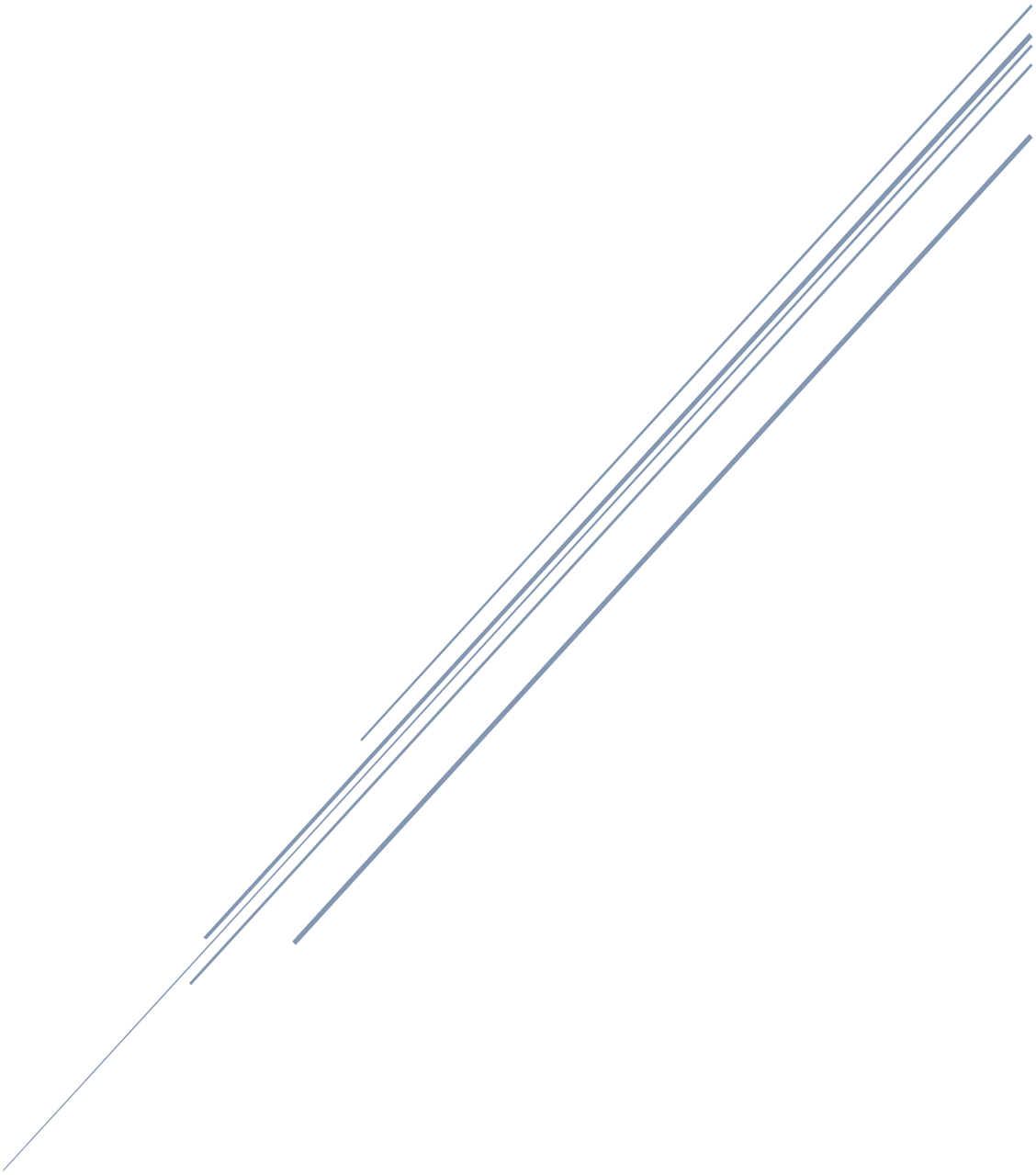
BINGO

Procedural Programming Project



Krisztian Nagy – G00330024
Software Development Year 2

Table of Contents

Introduction

The application is written for Procedural Programming module in year 2 of Software Development course at Galway Mayo Institute of Technology, Galway Campus.

The application is a BINGO (UK version) simulator that can simulate a game from beginning until end for minimum 2 and maximum 6 people.

Game

Application features

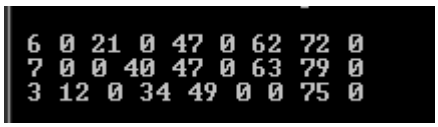
- Start new game
  - Choose the number of players between 2 and 6
- Load game
  - Load any previously saved game
- Exit

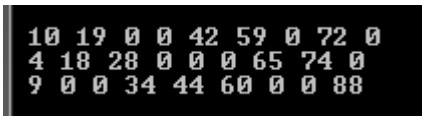
Game features

- Draw the next number
- Save game
  - If the game was opened from a previous save or the current game was saved once then two extra options are given
    - Over write the previous file
    - Create a new one
- Output the game status
  - Outputs the already drawn numbers in their obtained order and ascending order
  - Outputs each players card and the amount of numbers needed for each type of win on each card
- Exit Without save
  - Goes back to the main menu

Game Cards

Each player gets a game card. The card contains fifteen numbers distributed in three lines and nine columns. The cards have twelve empty spaces that is marked with zeros in the game. These spaces are distributed on the card as four space each row. A space can be in a corner as well.





Game Rules

The game has four winning types:

- Four corners
  - The leftmost and rightmost numbers on the top and bottom lines.
  - This winning type is not available on a card if any of the corner is empty.
  - The project instructions did not state and I was not able to find references about four corners win rule, I decided to develop this way.
- Line
  - Covering a horizontal line of five numbers on the ticket.
- Two Lines
  - Covering any two lines on the same ticket.
- Full House
  - Covering all fifteen numbers on the ticket.
  - Once a card has Full House win, then the game is finished.

Application design

Structure

The application is designed to be modular, therefore every main feature is separated into its own header and source file. The distribution of modules as follows:

Config

Enum of configuration

- Largest number on the card (equivalent of the amount of the numbers on card)
- Minimum number of cards
- Maximum number of cards
- Number of rows on card
- Number of columns on card
- Number of spaces in each row
- Maximum amount of numbers that drawable (equivalent of the largest number on the card)

Card

- The card structure is in the header file
- Card creation
- Random number generation
- Spaces generation
- Restriction of reoccurring numbers

Draw

- Draw a number

Rule

- Check for each rule on the cards
- Update the winning status on the cards

SaveLoad

- File_list structure.
  - It is used for a linked list that holds the file names that can be loaded as a game.
- Scan a folder for save game file that has bng extension
- Read drawn numbers from a file
- Read the number of cards and the cards from a file
- Save a game
  - Create a new file or over write an existing one
  - Dump the drawn numbers, number of cards and the cards into a file

Ui

- Everything what ever gets printed onto the screen comes from this module
- Get number of players
- Print cards to the screen
- Print drawn numbers to the screen
- Print card statistics to the screen
- Print the menu and the sub menus
  - Validates if the given number as menu item is an actual menu item
  - Keeps in a loop unless the valid menu item is chosen
- Show save a load status

Tinydir

- This module is not written by me. It is a freely available header for opening directories and files. It is available at https://github.com/cxong/tinydir

Main

- The main file pulling the menus together with the sub menus.
- The menu is built up with nested loops and combinations of switches.
- The main menu is running in an endless loop until the runApplicaton flag turs to false.
- Each menu point can be used in any order.
  - For example: Load a previous game then start a new game or the other way around. The flags are changing per required menu.

Platform dependency

The application was developed and tested on Windows 7. It has not been tested on any other operating systems and platforms therefoer it cannot be guaranteed to run on any other operating systems or platforms.

Obtained skills during development

Although the project development time is only nineteen hours, I used all the knowledge what we learned during class and lab hours in college. This knowledge was not enough to fully develop this application as I imagined, therefore I had to learn new processes and features of C programming language. I managed to learn how to create and use linked lists, integrate a third-party source and how to create header files to tidy up the code.