**Program 5**

CS2010 25 pts

Spring, 2018 Due: 11:59pm, Wednesday, March 21, 2018

**Problem:** Write a C++ program to play a modified version of the children's game Candy Land. Assume this game of Candy Land is played on a board with 85 squares. Your program should move one player from the starting square (square 1) to the Candy Castle square (square 85). The player starts at position 1 and draws a card to determine the number of squares to move on each turn until the player lands on exactly square 85. Card values are from 1 to 5. "Candy Bonus" moves exist at various points on the board to enhance play. If a player's token lands on a Candy Bonus square, the player immediately moves forward to the indicated square. The beginning and ending locations of each are:

**Candy Bonuses**

**If you land on square** **Jump to square**

6 (Rainbow Trail) 40

9 (Lord Licorice) 24

32 (Gumdrop Pass) 43

46 (Mr. Mint) 58

62 (Queen Frostine) 73

If a move would take the player beyond square 85, then the player's position should not be changed. The player must wait until the next turn to try to land exactly on square 85. The output should be a table with one line displayed for each turn indicating the move number, the player's current position, the value of the card drawn, the player's new position and if the player landed on a "Candy Bonus" square, the name of the square. A header for the table should be printed before the game begins.

**Processing:** Your main function should do the following:

1. Declare any variables needed, set the player's starting position to 1.

2. Call a function to display the heading for the table of player's moves on each turn.

3. Use a loop that takes a turn for the player each time through the loop until the player reaches square 85 exactly. A player's turn involves calling three functions in the body of the loop.

a. drawCard – "draws" a card and returns the card's value

b. movePlayer – given a card value from function drawCard, this function calculates the square on which the player will land (i.e., "moves" the player along the board). It passes back the new location for the player and the name of the "Candy Bonus" square if the player landed on one.

c. displayMove – displays a line for one turn showing the move number, current position, card number, new position and Candy Bonus name, if any.

*main*

*displayMove*

*movePlayer*

*drawCard*

*displayHeading*

Work on one function at a time. For each function decide what its parameter(s) should be: what data, if any, needs to be passed to the function; what results, if any, will the function return or pass back. Then write the function prototype, function call and function definition. Make sure these match in the required ways. Determine where each of these should be located in your program and type them in. Be sure to test and debug each function before going on to the next one.

**drawCard function:**

Use the following code for the *drawCard* function to get a card value from 1 to 5. You must add the line **#include <cstdlib>** at the beginning of your program to use the **rand** function. The **rand** function returns a randomly selected number each time it is called.

int drawCard()

{

return rand() % 5 + 1;

}

You will get the same series of numbers each time you run your program. This will help while you are developing and testing your program since the series of moves will be predictable. To makeyour program more interesting, once it is completely debugged and running correctly you can add the following line as the **first line** in your main function.

srand(time(NULL));

The **srand** function “seeds” the **rand** function with the system time so that you will get a different series of random numbers each time you run your program. When you start to use **srand**, you will also need to add the line **#include <ctime>** at the beginning of your program.

**Input:** No keyboard or file input is used for this program. The **rand** function is used to automatically generate a random series of numbers from 1 to 5 to represent the values on cards that would be drawn by a player in the real game.

**Sample output:**

**Candy Land Game**

**Your name**

**CS2010, Class time: xx:xx**

Move # Current Position Card New Position Candy Bonus

-------------------------------------------------------------------------------------

1 1 2 3

-------------------------------------------------------------------------------------

2 3 3 40 Rainbow Trail

-------------------------------------------------------------------------------------

3 40 5 45

-------------------------------------------------------------------------------------

etc.

**Program Documentation & Style:** Follow the same directions as on previous programs for the documentation. In addition, include function comments. Before the beginning of each function definition, include a sentence or two explaining the purpose of the function.

**Constant, variable declarations:** No global variable declarations may be used. Only constants may be declared globally.

**Turn in your .cpp file on Canvas.** Partial credit is given for any part of your program that is done so turn in your .cpp file even if you don't feel your program is complete or working correctly.

### CS2010 Program 5: Grading Rubric

**Due:** 11:59pm, Wednesday, March 21, 2018

\_\_\_\_\_\_\_ Project named correctly and .cpp file turned in on Canvas.

\_\_\_\_\_\_\_ (2) Header comments complete, including labeled sections identifying the Purpose, Input, Processing and Output

\_\_\_\_\_\_\_ (3) Function, in-line comments complete, uses good program style (meaningful datanames, whitespace, indentation)

\_\_\_\_\_\_\_ (3) *main* function calls function printHeading to display title, your name, column headings at beginning of table, data aligned neatly in columns. Uses parameter passing correctly. Function prototype included.

\_\_\_\_\_\_\_ (6) Loop set up for player to take a turn until player lands exactly on last square. Calls each function as described.

\_\_\_\_\_\_\_ (8) Functions used correctly to figure player's move (drawCard, movePlayer). Uses parameter passing correctly. Values returned are correct. Function prototypes included.

\_\_\_\_\_\_\_ (3) Function displayMove shows correct values for each move. Uses parameter passing correctly. Function prototype included.

\_\_\_\_\_\_\_ **(25) Total Points**