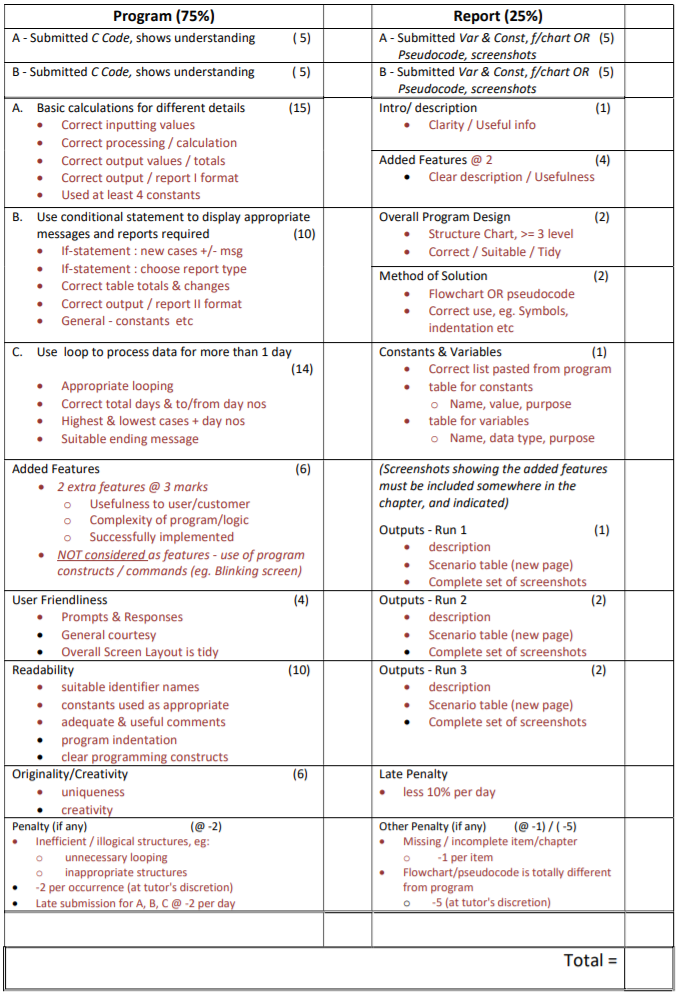


**AACS1074 PROGRAMMING CONCEPTS AND DESIGN I**

Assignment 2020/2021

|  |  |
| --- | --- |
| **Faculty** | Faculty of Computing and Information Technology (FOCS) |
| **Programme** | DCO – Diploma in Science (Computer Science and Computer Mathematics) Year 1 Semester 1 |
| **Course Code** | AACS1074 |
| **Course Title** | Programming Concepts and Design I |
| **Tutorial Group** | Group 1 |
| **Tutor Name** | Mr. Tan Cheng Tien |

|  |  |
| --- | --- |
| Student Name | Thong So Xue |
| Date Submitted | 4 September 2020 |
| Assignment Title | Assignment 2020/2021 - Covid-19 Cases Tracking System |

****

****

**About this Program**

The C source code of this program is located in the file named “Thong So Xue.c”, and it is submitted via Google Classroom to my tutor, Mr. Tan Cheng Tien. The source code can also be found at the appendix of this document.

I’m the author of this program, Thong So Xue. This program is created as an assignment for the course AACS1074 Programming Concepts and Design I, has been planned since the 20th of July, 2020, and was finally finished on the 23rd of August, 2020.

**Table of Content**

|  |  |
| --- | --- |
| Subchapter Heading | PAGE NUMBER |
| Front Cover | 1 |
| Assessment Criteria and Feedback Form | 2 |
| About this Program | 3 |
| Declaration of Originality | 4 |
| Table of Content | 5 |
| 1.0 Brief Description / Purpose of the Program | 6 |
| 2.0 Added Features | 7 |
| 3.0 Overall Program Design | 8 |
| 4.0 Method of Solution | 10 |
| 5.0 Constants & Variables | 17 |
| 5.1 Declaration & Variables | 17 |
| 5.2 Constants | 18 |
| 5.3 Variables | 19 |
| 6.0 Program Testing & Outputs | 22 |
| 6.1 Run 1 Scenario | 22 |
| 6.2 Run 2 Scenario | 30 |
| 6.3 Run 3 Scenario | 39 |
| 6.4 Presentation of Extra Features | 48 |
| 6.4.1 Title Screen | 48 |
| 6.4.2 Input Validation | 50 |
| 6.4.3 View Reports Repeatedly | 52 |
| 6.4.4 Danger Level Section | 53 |
| 6.4.5 Quit to Title Screen | 54 |
| 7.0 Appendix / Program Listing | 56 |

**1.0 Brief Description / Purpose**

During the COVID-19 pandemic, it is necessary for the local authority to efficiently keep track of all data regarding the disease cases. This program is a basic COVID-19 Cases Tracking System which is programmed to help the local authority to store every day’s cases data efficiently, and deliver Daily Reports or an overall summary respectively for analysis at the end of every day or session.

Every evening, the local authority will collect details of COVID-19 cases from all its district. Daily details include the number of new local and imported cases, number of deaths and number of recoveries. These daily details will then be the input for the program to further process it.

Before being able to input the daily details into the program, the program will first display an intro screen followed by the latest totals in a table. The user will then be asked to key in the current day’s inputs, which are the daily details mentioned above.

The program will then state how much the number of new cases has changed from yesterday by displaying a message, and then be able to produce two types of daily reports based on the inputted details and previous data. The user is able to choose between viewing the two reports, namely the Daily Summary Report and the Daily Comparative Report. The Daily Summary Report will show the total number of cases, total deaths, total recovered, total under treatment and their respective percentages, which are the death rate, recovery rate and under treatment rate. On the other hand, the Daily Comparative Report will show today’s and yesterday’s total number of new cases, number of total cases, death rate, recovery rate, and their respective differences compared between these two days.

After that, the user will be able to choose to continue inputting the data for the next day and so on, or choose to stop here. The process will repeat for the next day if the user wishes to continue, or the user will be redirected to the ending section if the user wishes to quit. The program will display the total number of days recorded, together with the starting and ending Day Numbers, the highest and lowest number of daily cases and the day on which it occurred, and finally a pleasant ending message at the ending section.

1. **Added Features**

**2.1 Title Screen**

A title screen is shown on every startup and contains several menu options to be chosen by the user. These options are “Start Inputting”, “Show Previous day’s Summary” and “Exit Application”. “Start Inputting” will start the program, starting from getting input data. “Show Previous day’s Summary” will print the previous day’s summary table, and pressing any key after that will redirect the user back to the title screen. “Exit Application” does what it says, exit the program.

**2.2 Input validation**

All inputs are strictly validated and any invalid input will be rejected by asking the user to re-enter the input again until the user enters the correct input format. There are two functions to validate inputs, one for integers and another for characters. The validation for integer inputs will reject all non-integer inputs except ‘x’ and ‘X’, and also rejects inputs that are out of range such as negative numbers or those numbers that are not listed in the menu options when entering menu options. The validation for character inputs will reject all inputs that have more than one character, and only accepts ‘y’, ‘Y’, ‘n’, ‘N’, ‘x’ and ‘X’ to be entered.

**2.3 View reports repeatedly**

Upon reaching the printing reports section, a menu with 3 options will be displayed and the user is allowed to choose between 3 options, which are “1 – Daily Summary Report”, “2 – Daily Comparative Report” and “3 – Quit Viewing Reports”. The first two options allow the user to repeatedly and indefinitely view the “Daily Summary Report” or the “Daily Comparative Report” until the user chooses the third option, instead of only being able to view one type of report for each day.

**2.4 Danger Level Section**

A Danger Level section is displayed is the ending section, which shows the danger level depending on the latest recorded death rate. Danger levels vary from “Extremely Low”, “Remarkably Low”, “Very Low”, “Low”, “Low-medium”, “Medium”, “Medium-high”, “High”, “Very High”, “Remarkably High”, “Extremely High” and lastly “God Save the Earth”.

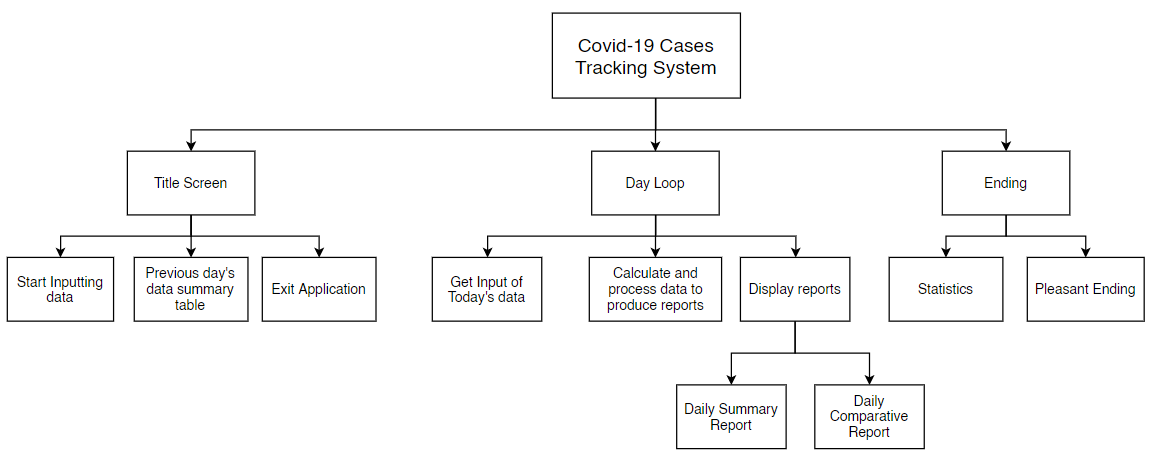
**2.5 Quit to title screen**

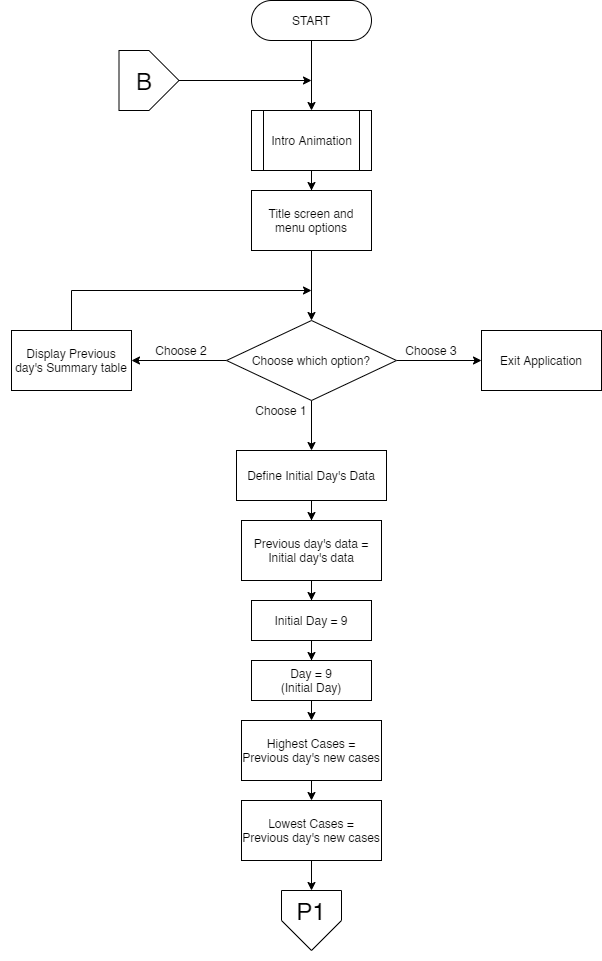
Allows the user to quit to the title at any momentby entering ‘x’ or ‘X’ at any input field. Upon entering ‘x’ or ‘X’ at any input field, the user will be asked again to double confirm if the user really wants to quit to the title screen, with a warning notifying the user that upon quitting to the title, any data that are not saved will be discarded. If the user wishes to quit by entering ‘y’ or ‘Y’, everything will be wiped out and the user is redirected back to the title screen. If the user does not want to quit and enters ‘n’ or ‘N’, the user will be brought back to where the user quits, which is the input field where the user had inputted ‘x’ or ‘X’.

Other than inputting ‘x’ or ‘X’, when the user reaches the ending section after choosing to quit when being asked to continue to the next day or not, a message stating that all data are saved will be displayed. Upon entering any key, the user will be brought back to the title screen. If the user chooses “Show Previous day’s Summary” at the title screen, what will be displayed is the summary report of the last day the user had inputted.

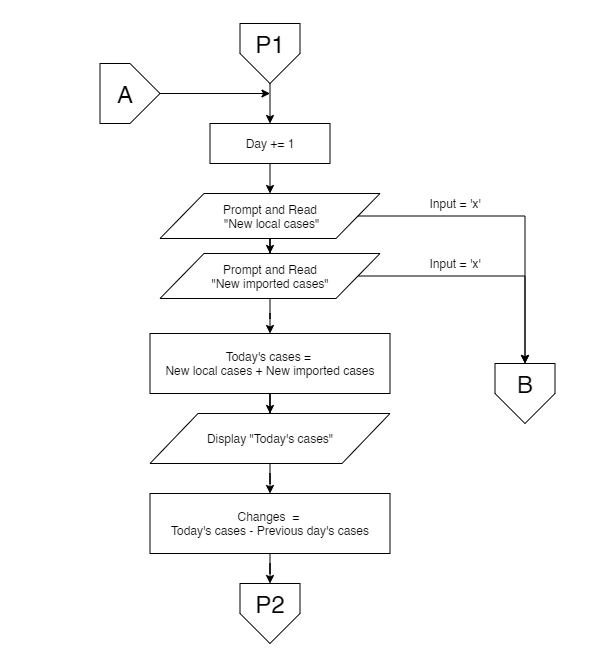
**3.0 Overall Program Design**

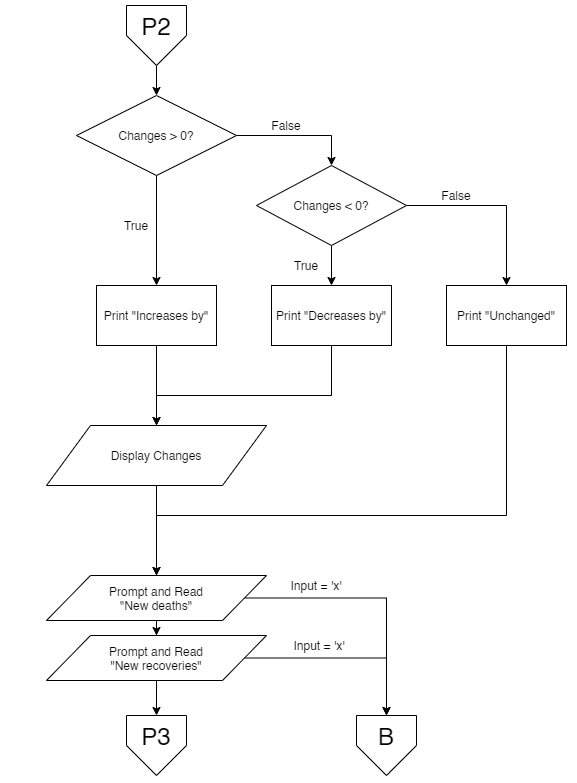
**Structure Chart**

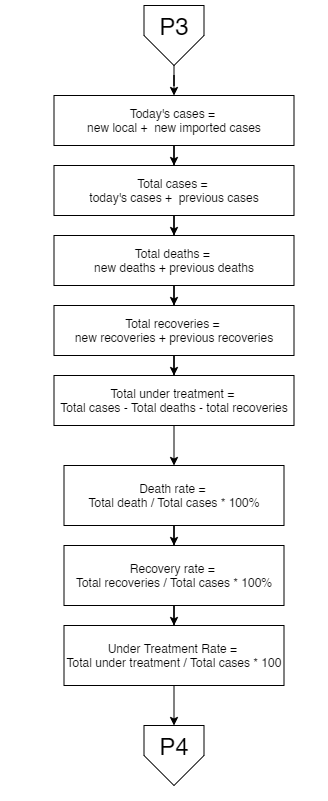
****

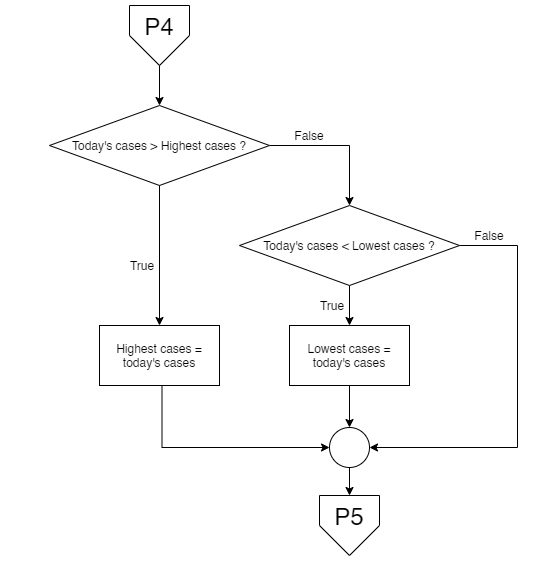
**4.0 Method of Solution**

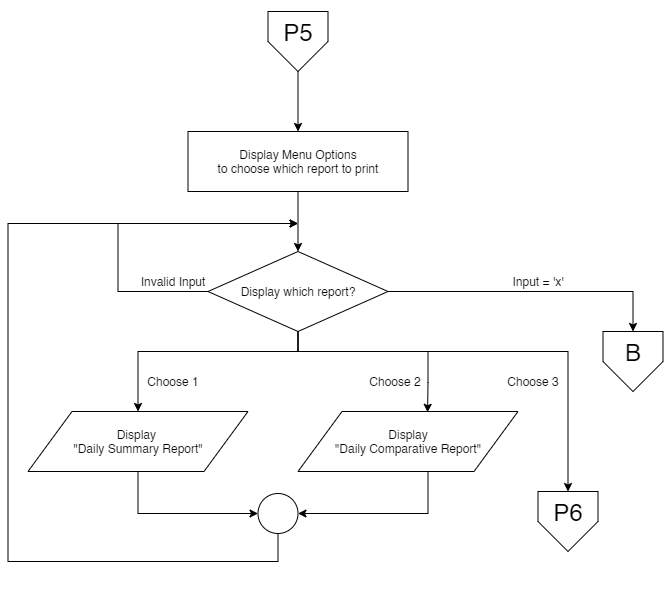
Flowchart

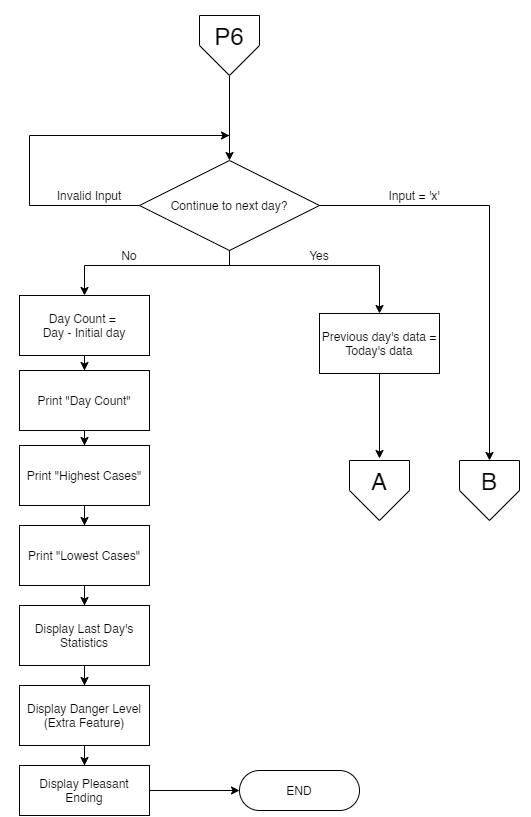
****

****

****

****

****

****

**5.0 Constants & Variables**

**5.1 Declaration and Definitions**

/\*==================== CONSTANTS ====================\*/

// Initial Day's Data

const int INI\_NEW\_CASES = 9;

const int INI\_TOTAL\_CASES = 100;

const int INI\_DEATHS = 2;

const int INI\_RECOVERED = 10;

const int INI\_DAY = 9;

// Page width fixed to 72

const int PAGE\_WIDTH = 72;

// Line breaks

const char LINE1[] = "--------------------------------------------------";

const char LINE2[] = "==================================================";

/\*==================== VARIABLES ====================\*/

// Data variables

int preNew, preTotal, preDeaths, preRecovered, preTreatment; // Previous day's data

float preDeathRate, preRecoveredRate, preTreatmentRate; // Previous day's rates

int newLocals, newImported, newTotal, newDeaths, newRecovered; // Today's data

int totalCases, totalDeaths, totalRecovered, totalTreatment; // Total of (All Previous + Today) data

float deathRate, recoveredRate, treatmentRate; // Today's rates

int newChanges; // Today's new cases - Previous day's new cases

int currentDay, daycount; // Day variables

int highest, lowest, highestDay, lowestDay; // The highest and lowest cases and their respective days

// Option selections

int menuSelect, reportSelect; // Get option selections for the title menu and the report menu

char contiLoopDay; // Get Option selection for "Continue to next day? (Y/N)"

// Report variables

int viewCount; // Counts how many times the user had viewed the report

// Title screen variables

int atTitle = 0; // Whether the user is at the title screen.

**5.2 Constants**

|  |  |  |
| --- | --- | --- |
| **Constant Name** | **Value** | **Purpose** |
| INI\_NEW\_CASES | 9 | Initial day or previous day (Day 9)’s number of new cases. |
| INI\_TOTAL\_CASES | 100 | Initial day or previous day (Day 9)’s number of total cases. |
| INI\_DEATHS | 2 | Initial day or previous day (Day 9)’s number of deaths. |
| INI\_RECOVERED | 10 | Initial day or previous day (Day 9)’s number of recoveries. |
| INI\_DAY | 9 | Initial day or previous day’s number (Day 9) |
| PAGE\_WIDTH | 72 | The total width of the whole program printing. This is used in center aligning texts. |
| LINE[50] | -------------------------------------------------- | Used as small line breaks and in tables |
| LINE2[50] | ================================================== | Used as large line breaks between days. |

**5.3 Variables**

|  |  |  |
| --- | --- | --- |
| **Data Type** | **Variable Name** | **Purpose** |
| integer | preNew | Stores the number of new cases in the previous day. INI\_NEW\_CASES will be assigned to this variable on program startup. When a user chooses to continue to the next day, the value of today’s data, newTotal will be assigned to this variable. |
| integer | preTotal | Stores the number of total cases up until the previous day. INI\_TOTAL\_CASES will be assigned to this variable on program startup. When a user chooses to continue to the next day, the value of today’s data, totalCases will be assigned to this variable. |
| integer | preDeaths | Stores the number of total deaths up until the previous day. INI\_DEATHS will be assigned to this variable on program startup. When a user chooses to continue to the next day, the value of today’s data, totalDeaths will be assigned to this variable. |
| integer | preRecovered | Stores the number of total recoveries up until the previous day. INI\_RECOVERED will be assigned to this variable on program startup. When a user chooses to continue to the next day, the value of today’s data, totalRecovered will be assigned to this variable. |
| integer | preTreatment | Stores the number of total patients under treatment up until the previous day. When a user chooses to continue to the next day, the value of today’s data, totalTreatment will be assigned to this variable. |
| Float | preDeathRate | Stores the total death rate up the previous day using the formula: (preDeathRate = (float)predeaths / preTotal \* 100). When a user chooses to continue to the next day, the value of today’s data, deathRate will be assigned to this variable. |
| Float | preRecoveredRate | Stores the total recovery rate up until the previous day using the formula: (preRecoveredRate = (float)preRecovered / preTotal \* 100). When a user chooses to continue to the next day, the value of today’s data, recoveredRate will be assigned to this variable. |
| Float | preTreatmentRate | Stores the total patients under treatment rate up until the previous day using the formula: (preTreatmentRate = (float)preTreatment / preTotal \* 100). When a user chooses to continue to the next day, the value of today’s data, treatmentRate will be assigned to this variable. |
| integer | newLocals | Stores the number of new local cases today inputted by the user. |
| integer | newImported | Stores the number of new imported cases today inputted by the user. |
| integer | newTotal | Stores the number of total new cases today, which is obtained using the formula: (newTotal = newLocals + newImported). |
| integer | newDeaths | Stores the number of deaths today inputted by the user. |
| integer | newRecovered | Stores the number of recoveries today inputted by the user. |
| integer | totalCases | Stores the number of total cases of all time. The value is obtained by using the formula: (totalCases = preTotal + newTotal). |
| integer | totalDeaths | Stores the number of total deaths of all time. The value is obtained by using the formula: (totalDeaths = preDeaths + newDeaths). |
| integer | totalRecovered | Stores the number of total recoveries of all time. The value is obtained by using the formula: (totalRecovered = preRecovered + newRecovered). |
| integer | totalTreatment | Stores the number of total patients under treatment of all time. The value is obtained by using the formula: (totalTreatment = preTreatment + newTreatment). |
| float | deathRate | Stores the total death rate of all time using the formula: (deathRate = (float)totalDeaths / totalCases \* 100). This is also used to determine the danger level at the ending section. |
| float | recoveredRate | Stores the total recovery rate of all time using the formula: (recoveredRate = (float)totalRecovered / totalCases \* 100). |
| float | treatmentRate | Stores the total patients under treatment rate up of all time using the formula: (treatmentRate = (float)totalTreatment / totalCases \* 100). |
| integer | newChanges | Stores the number of differences between the total number of new cases today, and the total number of new cases in the previous day. |
| integer | currentDay | Stores the current day’s day number. This is used at displaying the current day number at every start of new day, and is later used as statistics in the ending section. |
| integer | dayCount | Stores the number of days the user has inputted. This is later used as statistics in the ending section. |
| integer | highest | Stores the highest amount of daily cases recorded. This is later used as statistics in the ending section. |
| integer | lowest | Stores the lowest amount of daily cases recorded. This is later used as statistics in the ending section. |
| integer | highestDay | Stores the day that have the highest amount of daily cases recorded. This is later used as statistics in the ending section. |
| integer | lowestDay | Stores the day that have the lowest amount of daily cases recorded. This is later used as statistics in the ending section. |
| integer | menuSelect | Stores the user’s selection of the menu options in the title screen. |
| integer | reportSelect | Stores the user’s selection of the menu options to print which report or to quit printing reports. |
| character | contiLoopDay | Stores the user’s selection to whether continue to the next day or to stop here and proceed to the ending. |
| integer | viewCount | Stores how many times the user has viewed the report repeatedly in a day. This is used to determine whether should the program print the menu options for the printing reports section again. |
| integer | atTitle | Value changes depending on whether the user is at the title screen. 1 = The user is at the title screen, 0 = The user is not at the title screen. This value is used to disable quit to title function when the user is already at the title. |

**6.0 Program Testing & Outputs**

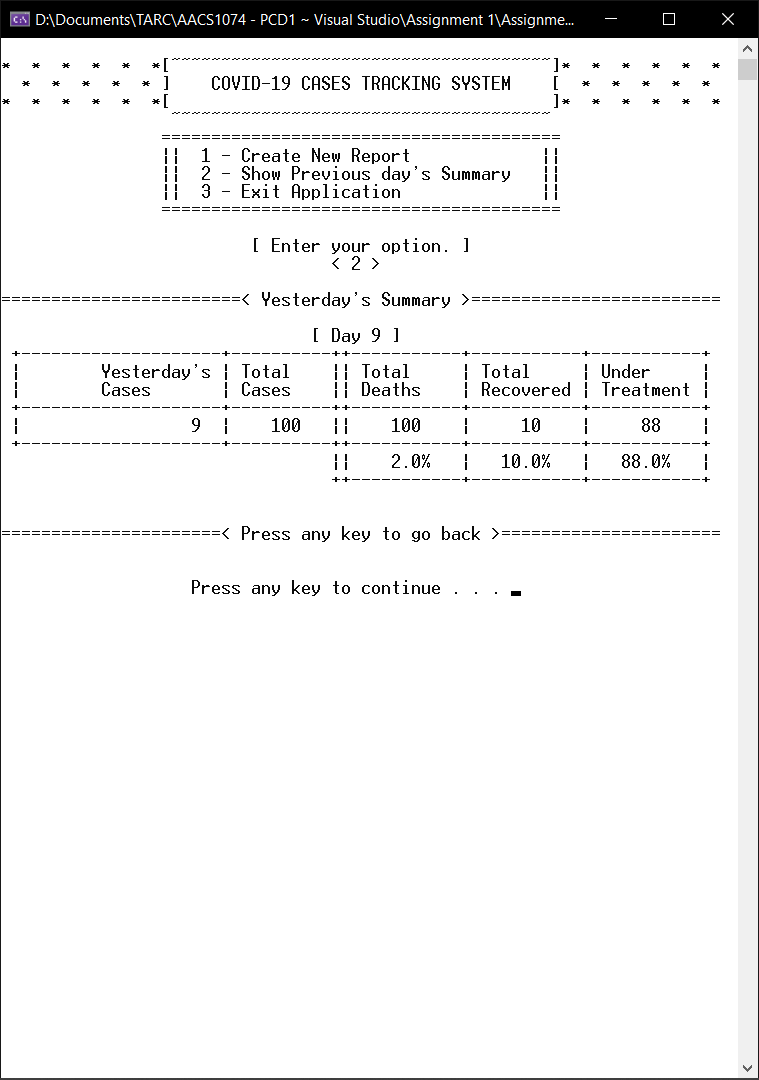
**6.1 Run 1 Scenario**

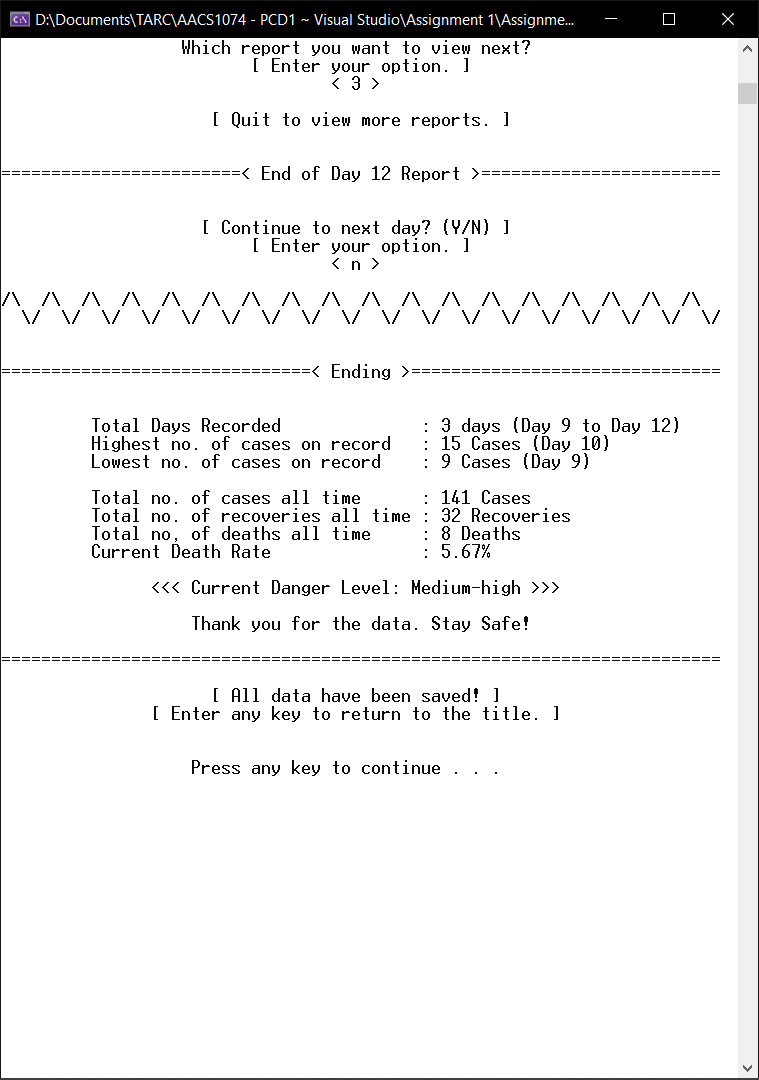
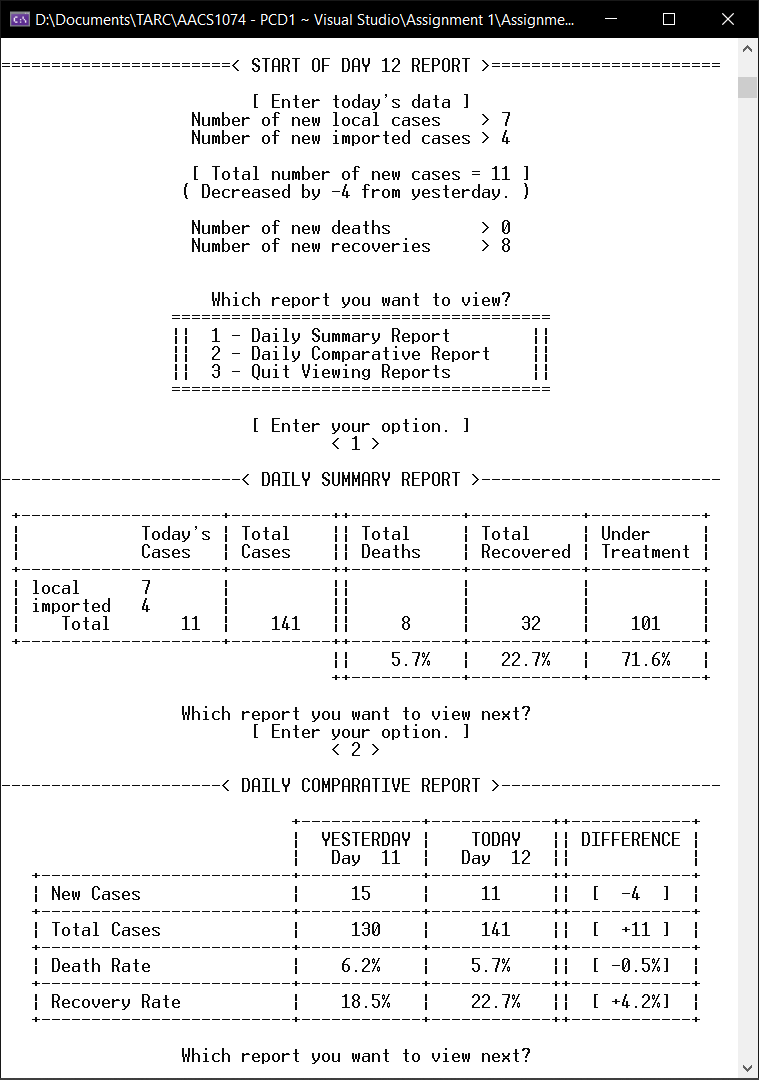
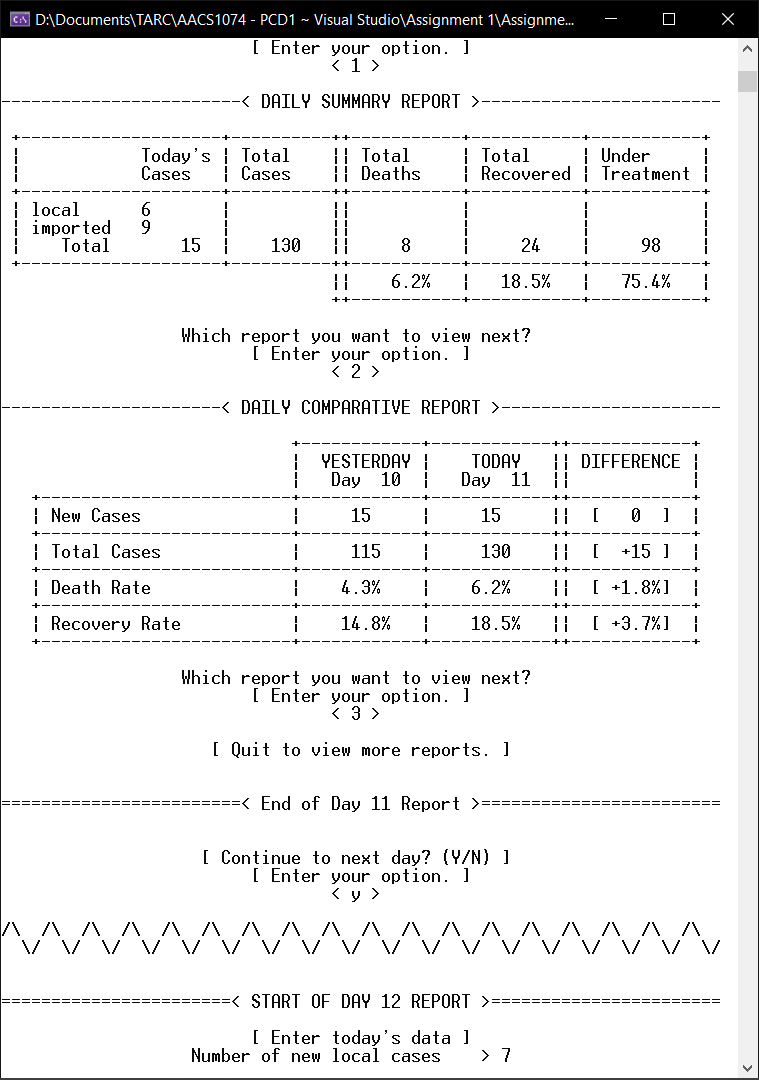
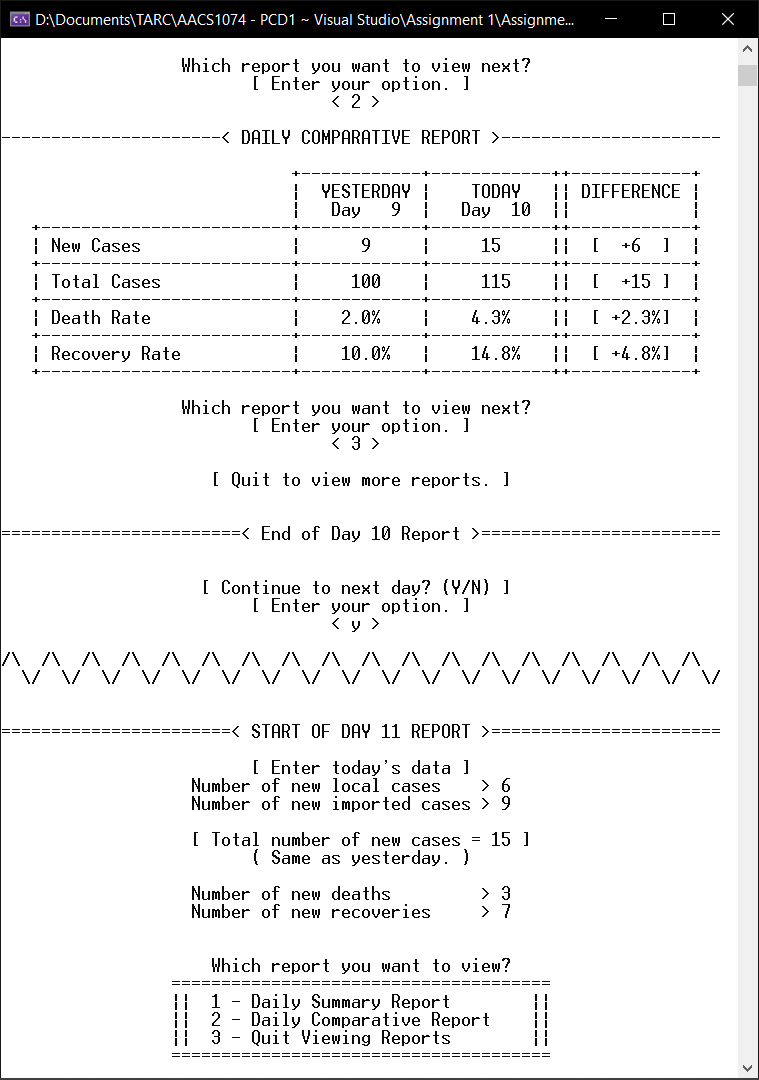
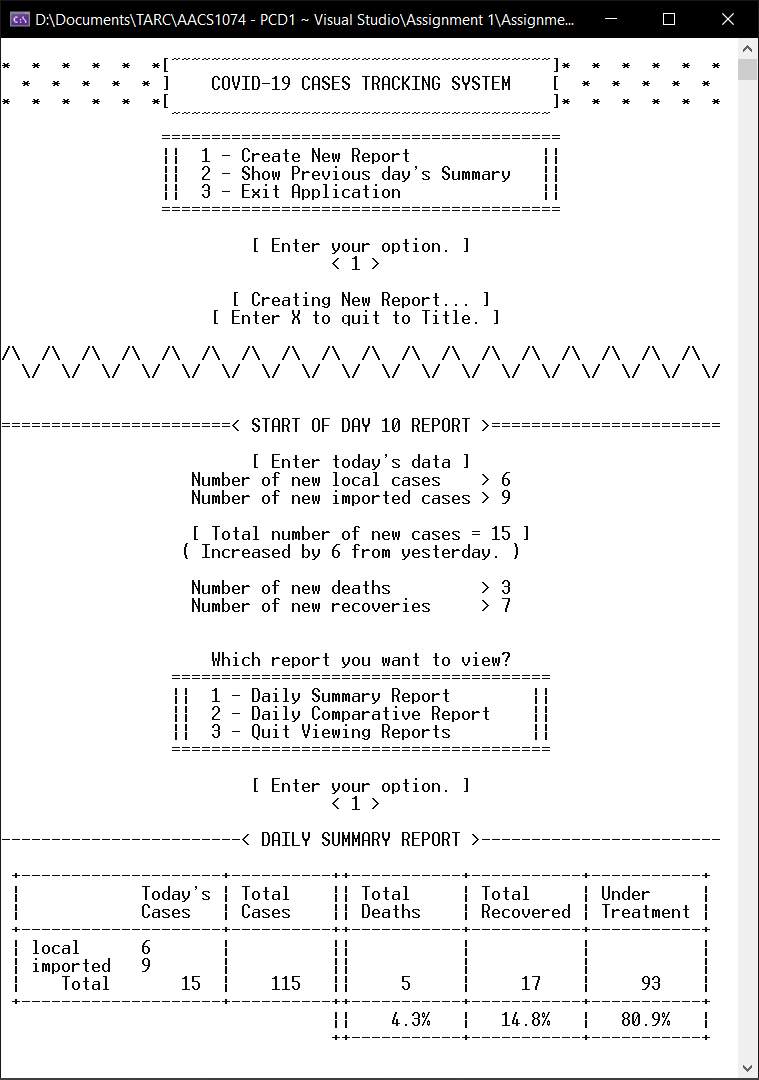
**Scenario Description**

3 days of input data, the first and second loop’s inputted data are the same. The third loop’s number of new cases is slightly less than the second loop, and its death count is also set to zero. The highest number of new cases should be at the first loop, even though the second loop has the same number of new cases. The lowest number of new cases should be at the initial day before the loop, which is Day 9 with 9 new cases.

**Test Data + Expected Outputs**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Inputs | | | | Expected Result / Outputs | | | | |
| Loop No. | Day No. | New Local Cases | New Imported Cases | No. of deaths | No. recovered | New cases | Total cases | Total Deaths | Total Recovered | Total under treatment |
|  | 9 |  |  |  |  | 9 | 100 | 2  (2.0%) | 10  (10.0%) | 88  (88.0%) |
| 1 | 10 | 6 | 9 | 3 | 7 | 15,  Increased by 6 | 100+15 =115 | 2+3=5  **(4.3%)**  change=  +2.3% | 10+7=17  **(14.8%)**  change =  +4.8% | 115-5-17 =93  **(80.9%)** |
| 2 | 11 | 6 | 9 | 3 | 7 | 15,  Same as yesterday | 115 +  15  = 130 | 5+3=8 **(6.2%)** change=  +1.8% | 17+7=24  **(18.5%)** change=  +3.7% | 130-8-24 =98  **(75.4%)** |
| 3 | 12 | 7 | 4 | 0 | 8 | 11,  Decreased by 4 | 130 +  11  = 141 | 8+0=8 **(5.7%)** change=  -0.5% | 24+8=32 **(22.7%)** change=  +4.2% | 141-8-32  =101  **(71.6%)** |
|  |  |  |  |  |  | Ending Message:  Total days recorded = 3 days (Day 10 to Day 12)  Highest daily cases = 15, on Day 10  Lowest daily cases = 9, on Day 9 | | | | |

**Screenshots**

****

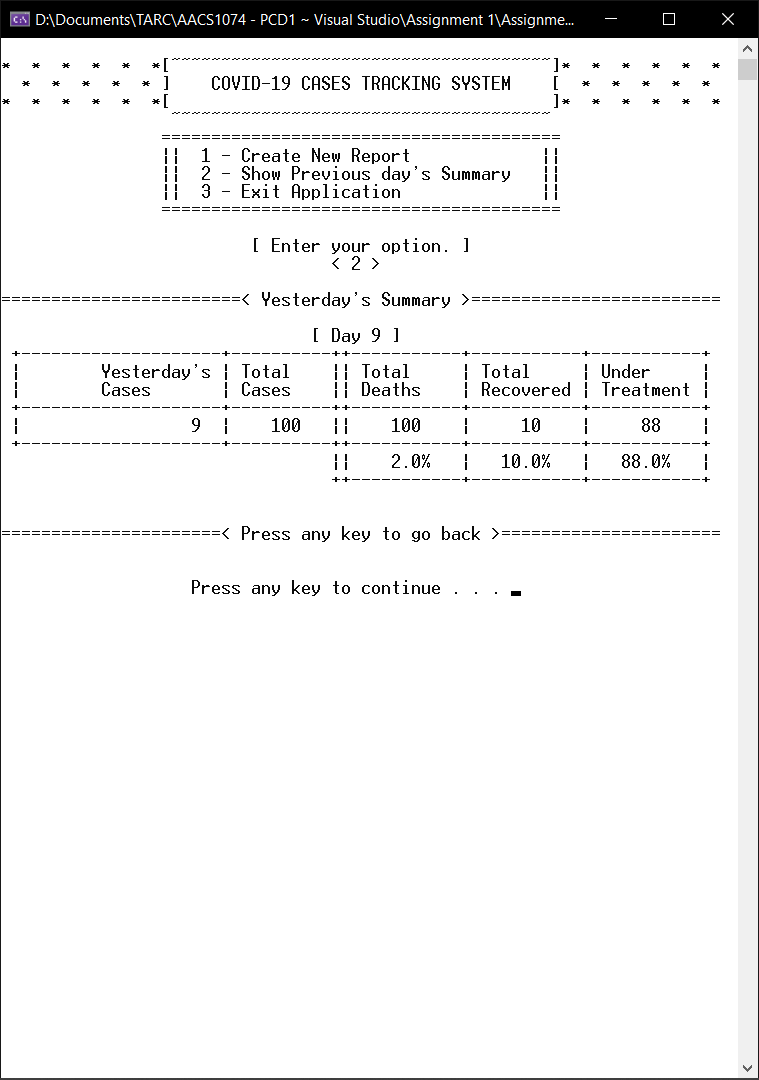
**6.2 Run 2 Scenario**

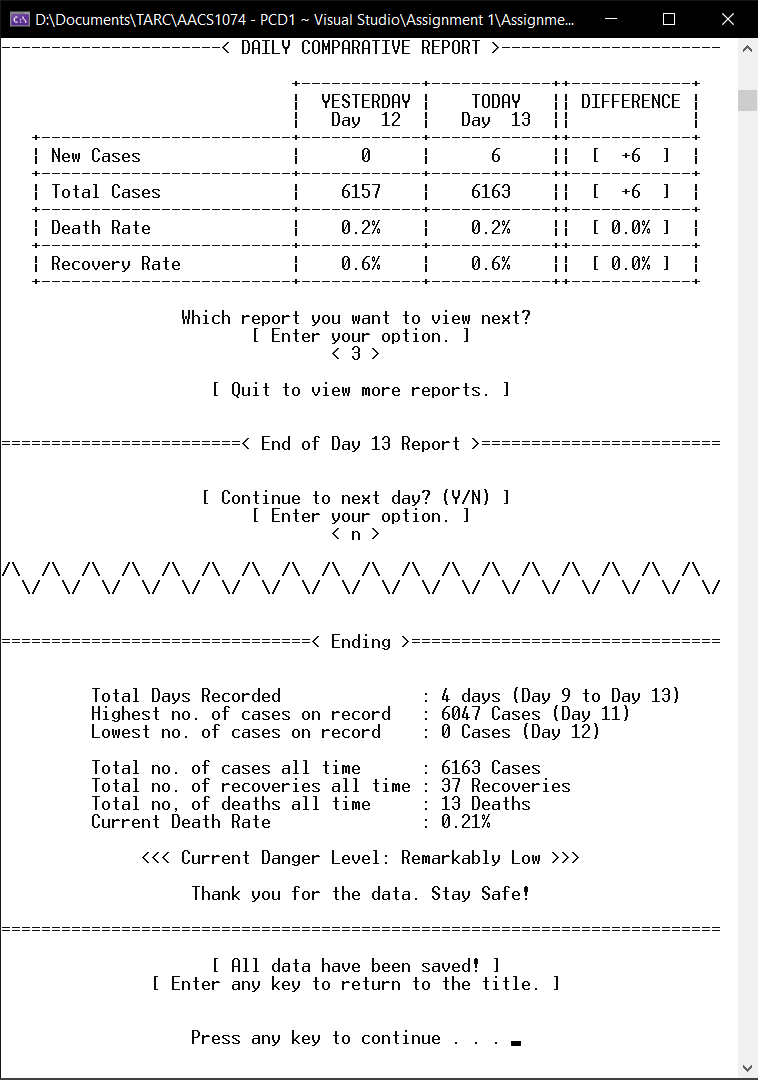
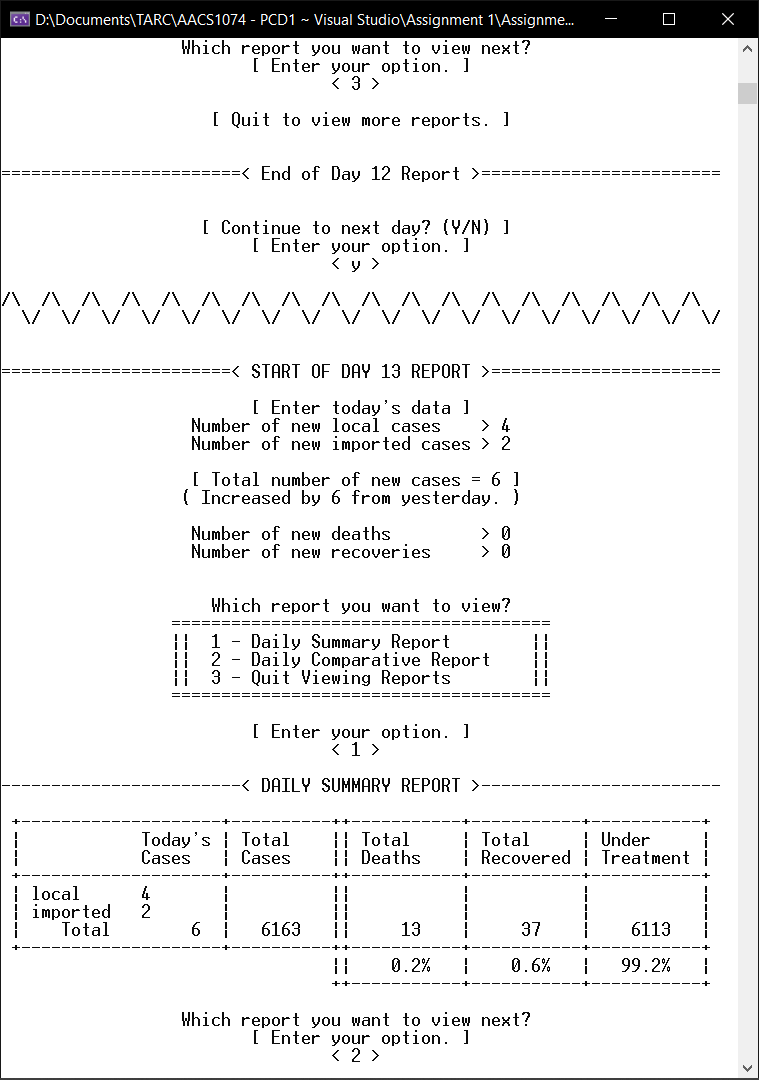
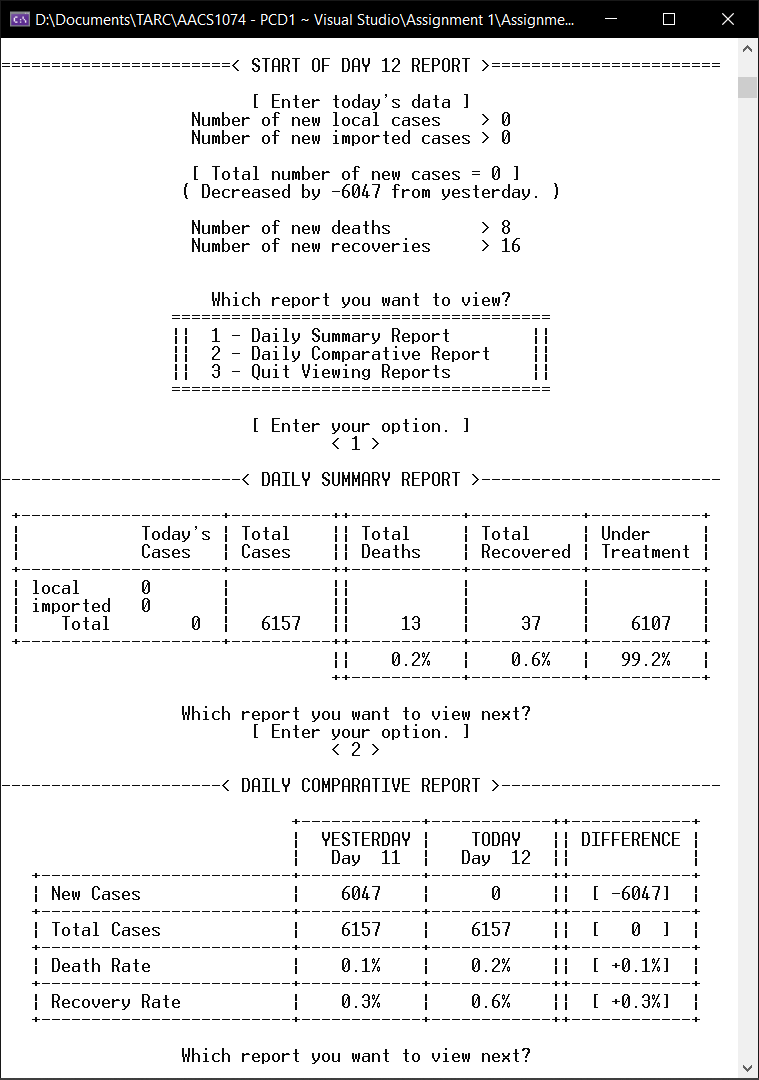
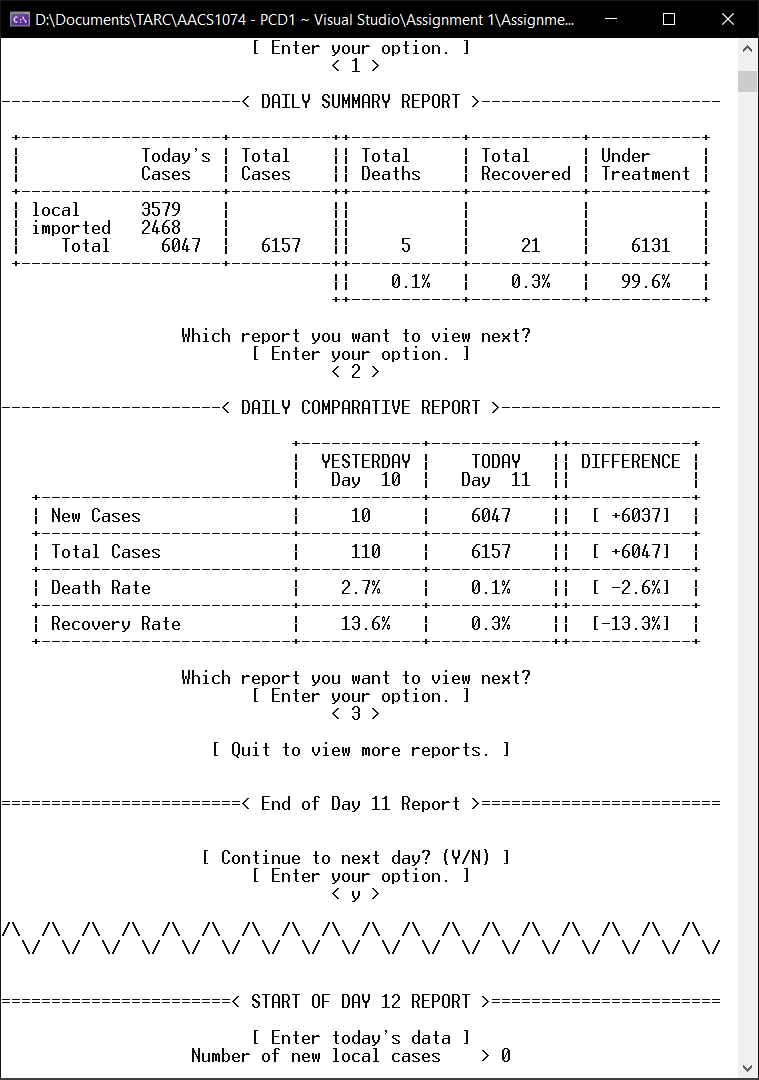
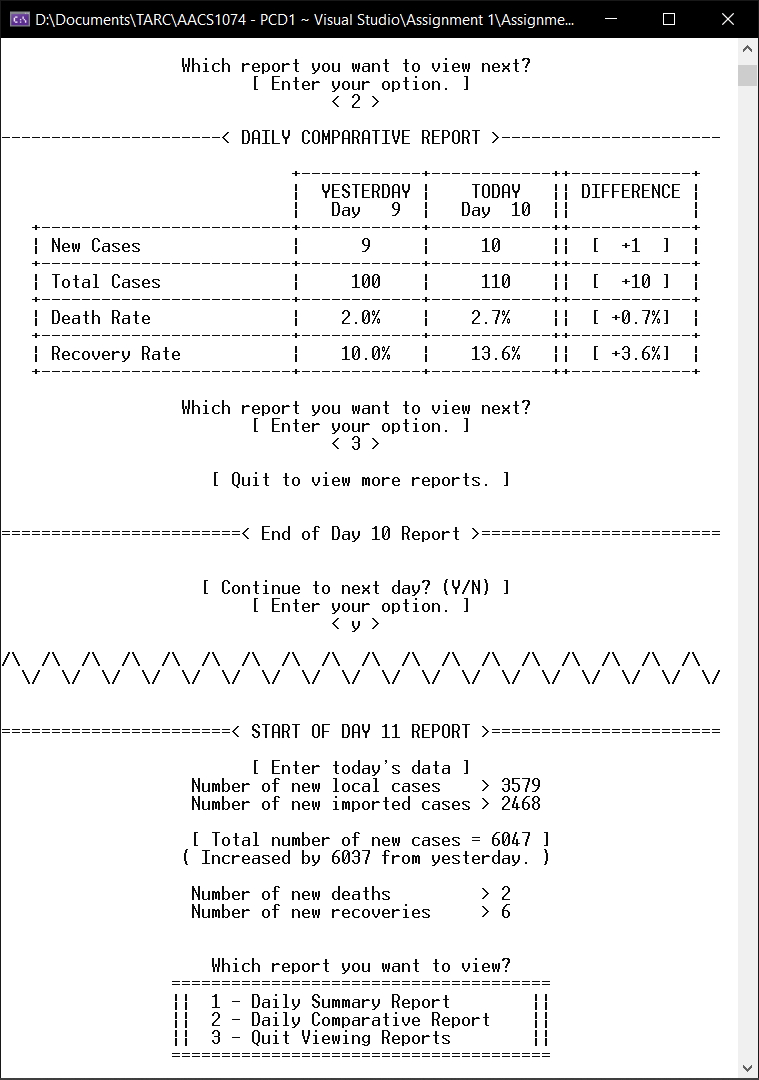
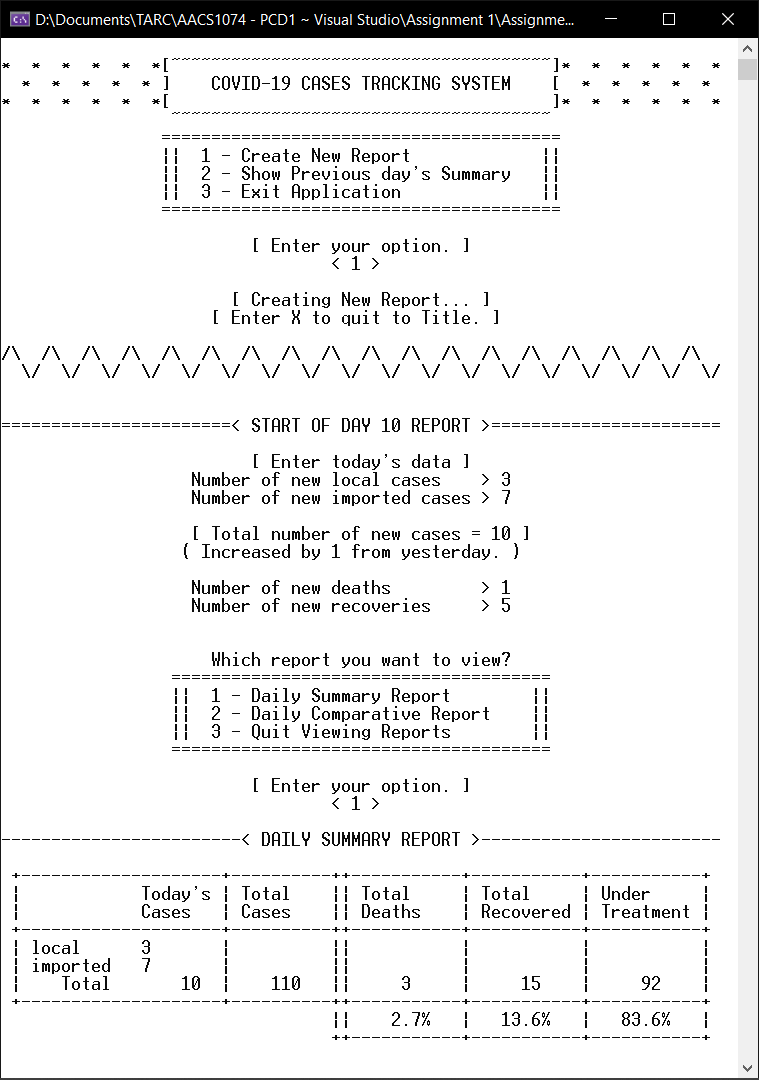
**Scenario Description**

4 days of input data. The second loop’s number of new cases is significantly higher than other loops. The third loop’s number of new cases is set to zero, and the fourth loop’s number of deaths and recoveries are set to zero. The highest number of new cases should be at the second loop with a large number, and the lowest should be at the third loop with a value of zero.

**Test Data + Expected Outputs**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Inputs | | | | Expected Result / Outputs | | | | |
| Loop No. | Day No. | New Local Cases | New Imported Cases | No. of deaths | No. recovered | New cases | Total cases | Total Deaths | Total Recovered | Total under treatment |
|  | 9 |  |  |  |  | 9 | 100 | 2  (2.0%) | 10  (10.0%) | 88  (88.0%) |
| 1 | 10 | 3 | 7 | 1 | 5 | 10,  Increase by 1 | 100+10 =110 | 2+1=3  **(2.7%)**  change=  +0.7% | 10+5=15  **(13.6%)**  change =  +3.6% | 110-3-15  = 92  **(83.6%)** |
| 2 | 11 | 3579 | 2468 | 2 | 6 | 6047, Increase by 6037 | 110+6047 =6157 | 3+2=5 **(0.1%)** change=  -2.6% | 15+6=21 **(0.3%)** change=  -13.3% | 6157-5-21  = 6131  **(99.6%)** |
| 3 | 12 | 0 | 0 | 8 | 16 | 0,  Decrease by 6047 | 6157+0 =6157 | 5+8=13 **(0.2%)** change=  +0.1% | 21+16=37 **(0.6%)** change=  +0.3% | 6157-13-37  = 6107  **(99.2%)** |
| 4 | 13 | 4 | 2 | 0 | 0 | 6,  Increase by 6 | 6157+6 =6163 | 13+0=13 **(0.2%)** change=  0.0% | 37+0=37 **(0.6%)** change=  0.0% | 6163-13-37  = 6113  **(99.2%)** |
|  |  |  |  |  |  | Ending Message:  Total days recorded = 4 days (Day 10 to Day 13)  Highest daily cases = 6157, on Day 11  Lowest daily cases = 0, on Day 12 | | | | |

**Screenshots**

****

**6.3 Run 3 Scenario**

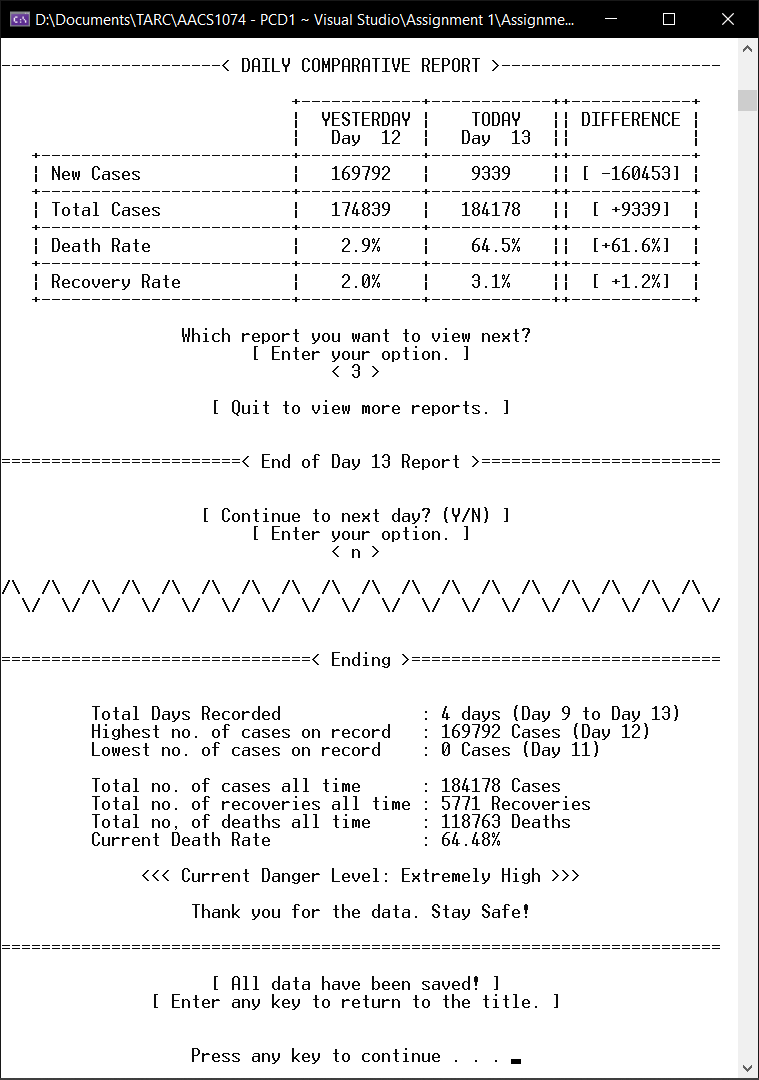
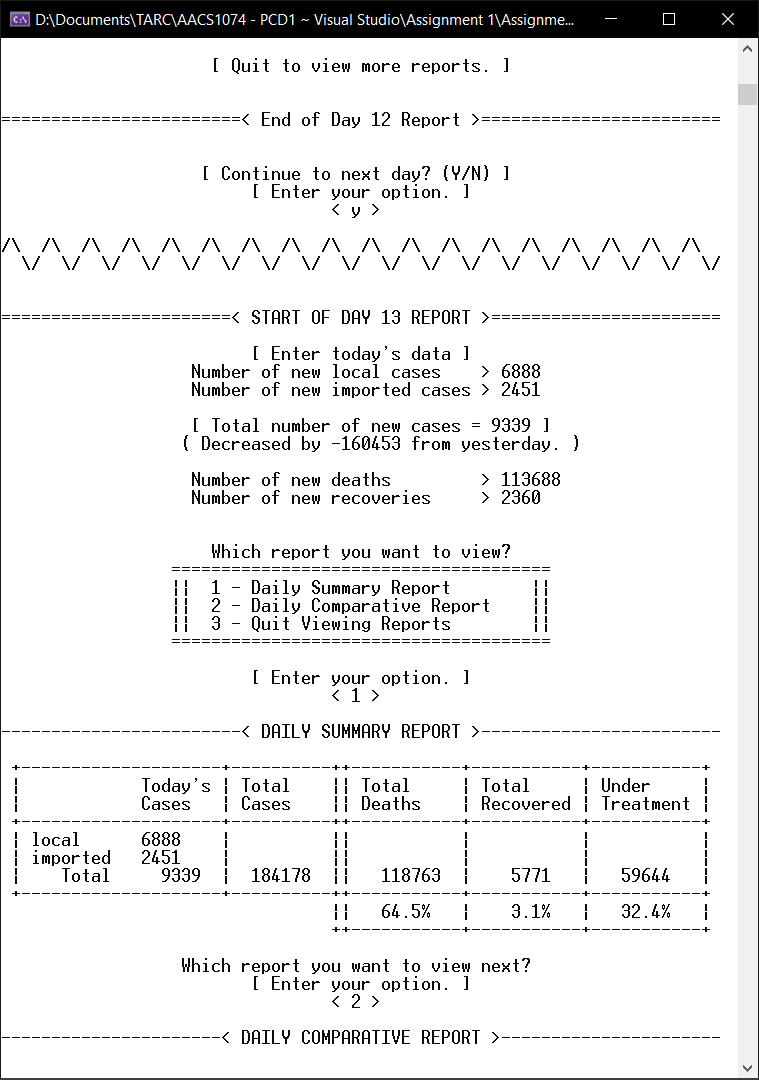
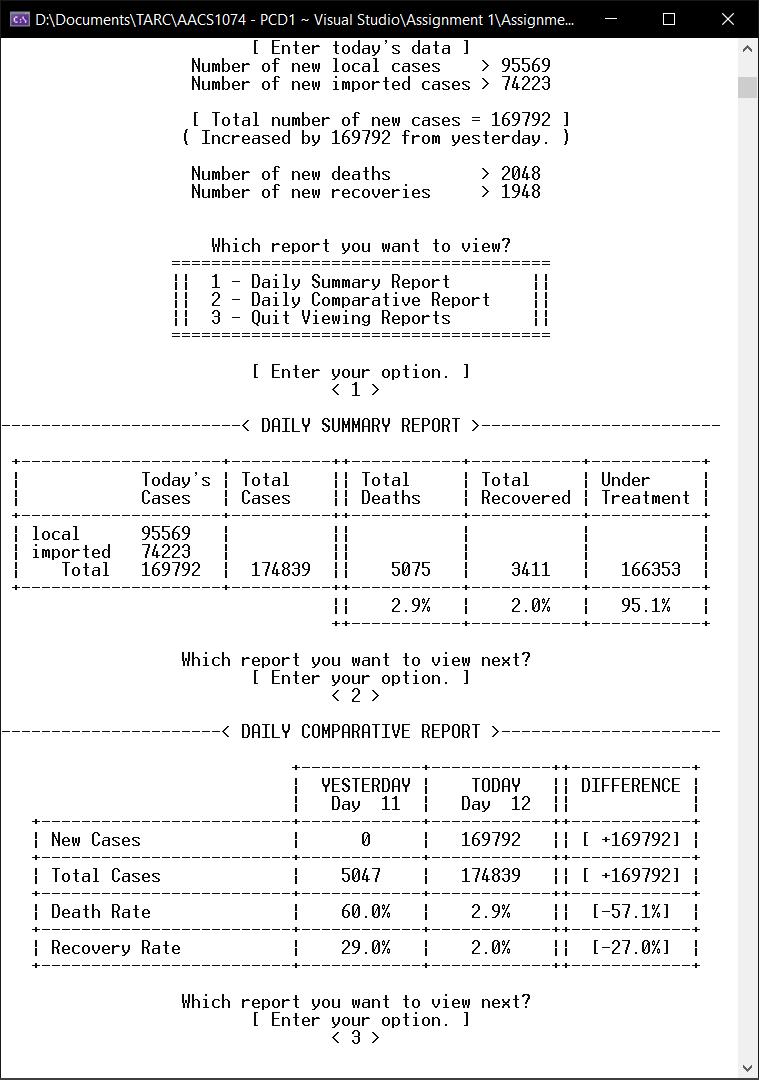
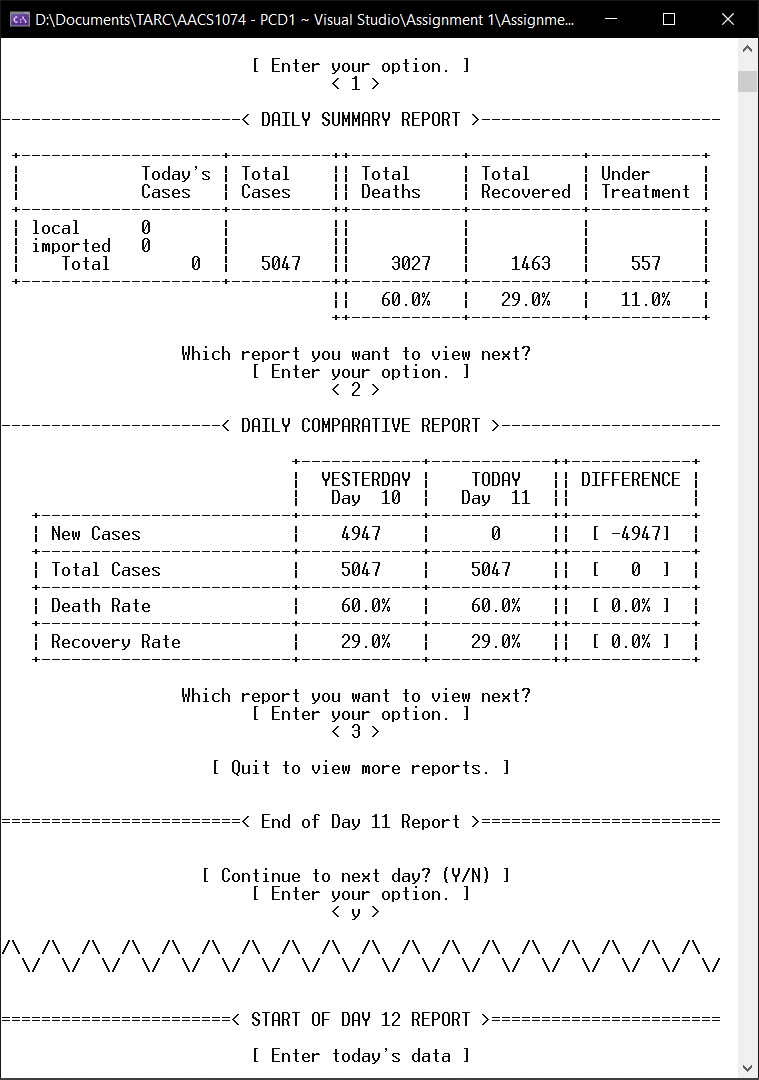
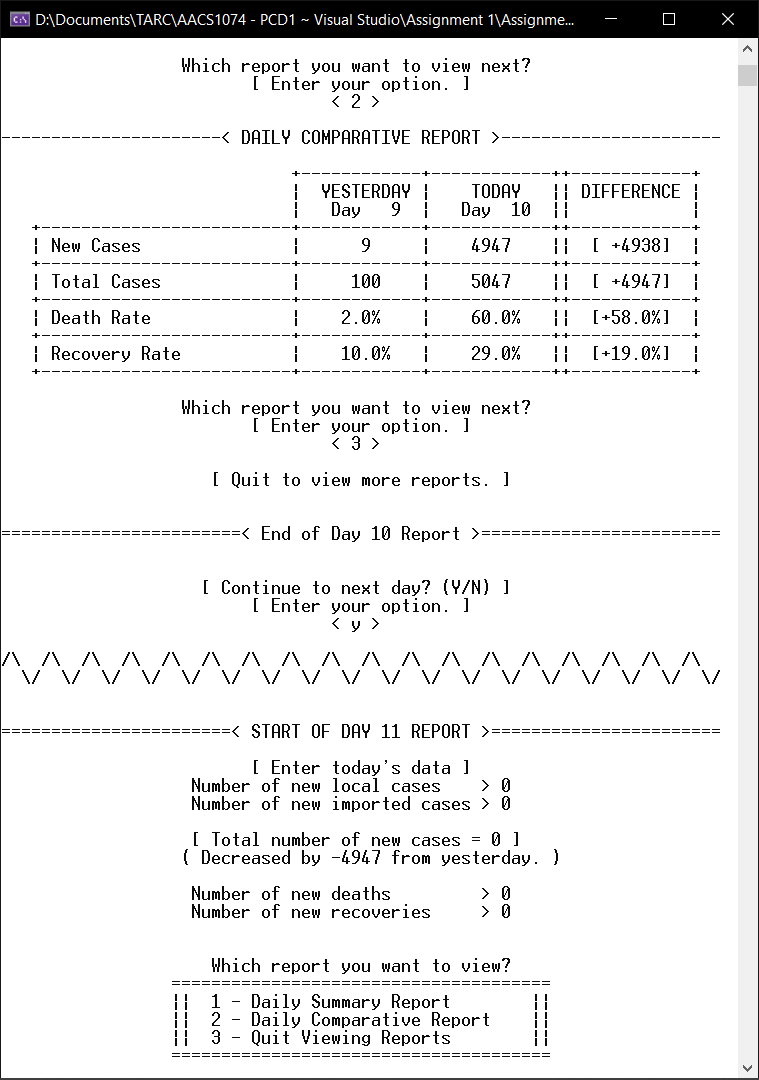
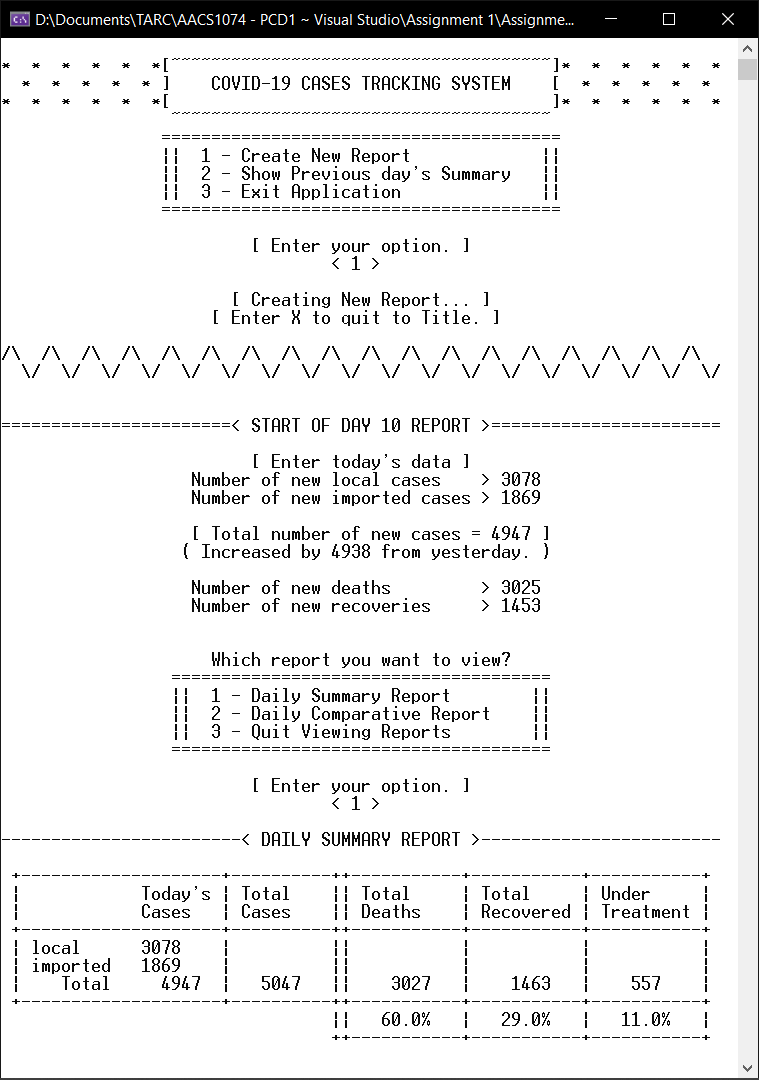
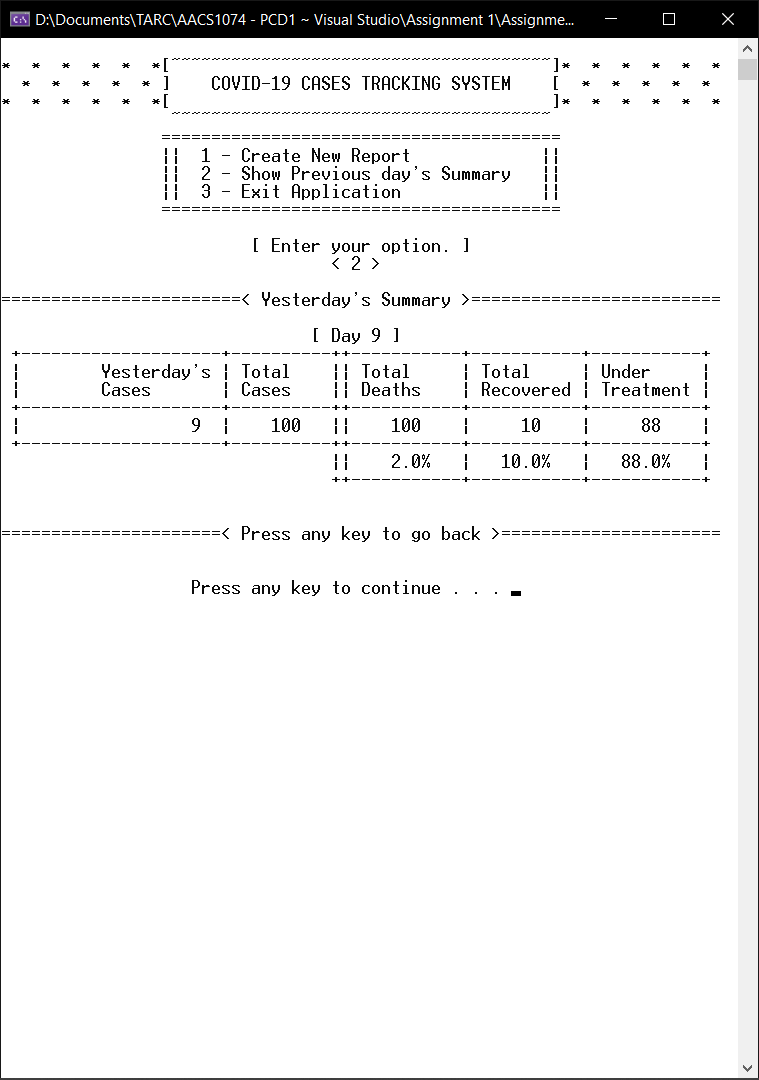
**Scenario Description**

4 days of input data. All input data are set to large numbers. The second loop’s number of new cases, deaths and recoveries are all set to zero. The third loop’s number of new cases is significantly higher than the other days. The fourth loop’s number of deaths is significantly higher than other days. The highest number of new cases is at the fourth loop, while the lowest should be at the second loop with a value of zero.

**Test Data + Expected Outputs**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Inputs | | | | Expected Result / Outputs | | | | |
| Loop No. | Day No. | New Local Cases | New Imported Cases | No. of deaths | No. of recoveries | New cases | Total cases | Total Deaths | Total Recovered | Total under treatment |
|  | 9 |  |  |  |  | 9 | 100 | 2  (2.0%) | 10  (10.0%) | 88  (88.0%) |
| 1 | 10 | 3078 | 1869 | 3025 | 1453 | 4947,  Increase  by 4938 | 100 +  4947  =5047 | 2+3025  =3027  **(60.0%)**  change=  +58.0% | 10+1453  =1463  **(29.0%)**  change =  +19.0% | 5047-3027-1463  = 557  **(11.0%)** |
| 2 | 11 | 0 | 0 | 0 | 0 | 0,  Decrease  by 4947 | 5047 +0  =5047 | 3027+0  =3027 **(60.0%)** change=  0.0% | 1463+0  =1463 **(29.0%)** change=  0.0% | 5047- 3027-1463  =557  **(11.0%)** |
| 3 | 12 | 95569 | 74223 | 2048 | 1948 | 169792,  Increase  by 169792 | 5047+ 169792  = 174839 | 3027 +2048 =5075 **(2.9%)** change=  -57.1% | 1463+1948 =3411 **(2.0%)** change=  -27.0% | 174839-5075-3411= 166353  **(95.1%)** |
| 4 | 13 | 6888 | 2451 | 113688 | 2360 | 9339,  Decrease  by 160453 | 174839 +9339 =184178 | 5075 +113688 =118763 **(64.5%)** change=  +61.6% | 3411+2360 =5771 **(3.1%)** change=  +1.2% | 184178-118763-5771 =59644  **(32.4%)** |
|  |  |  |  |  |  | Ending Message:  Total days recorded = 4 days (Day 10 to Day 13)  Highest daily cases = 169792, on Day 12  Lowest daily cases = 0, on Day 11 | | | | |

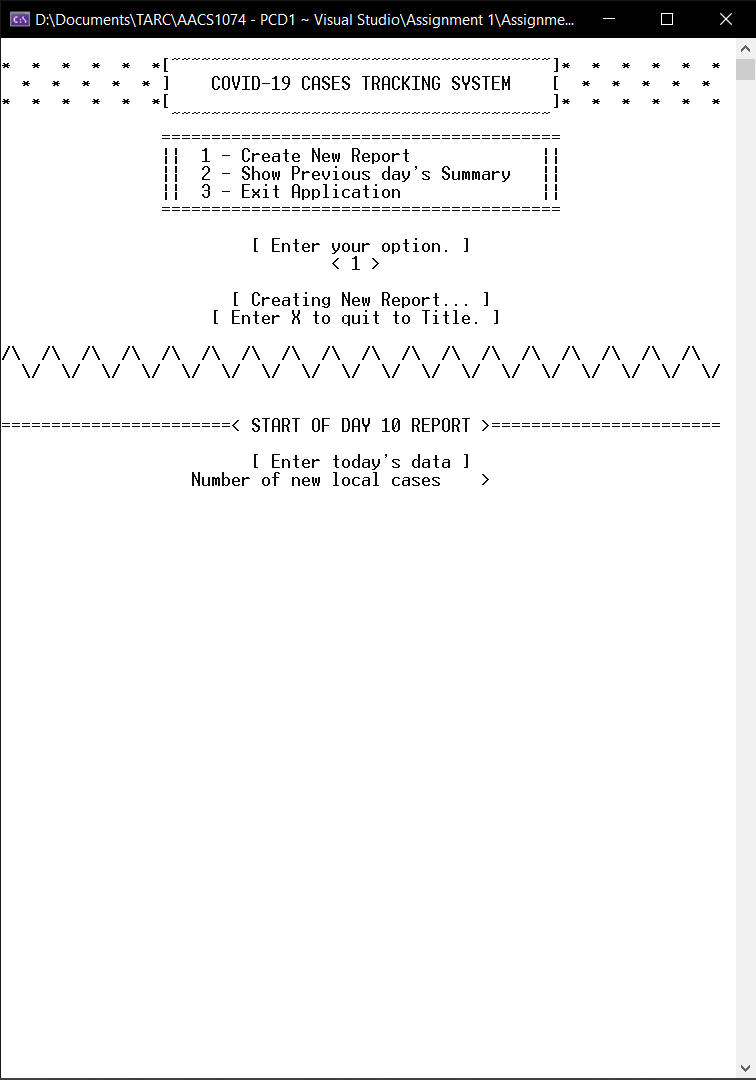
**Screenshots**

****

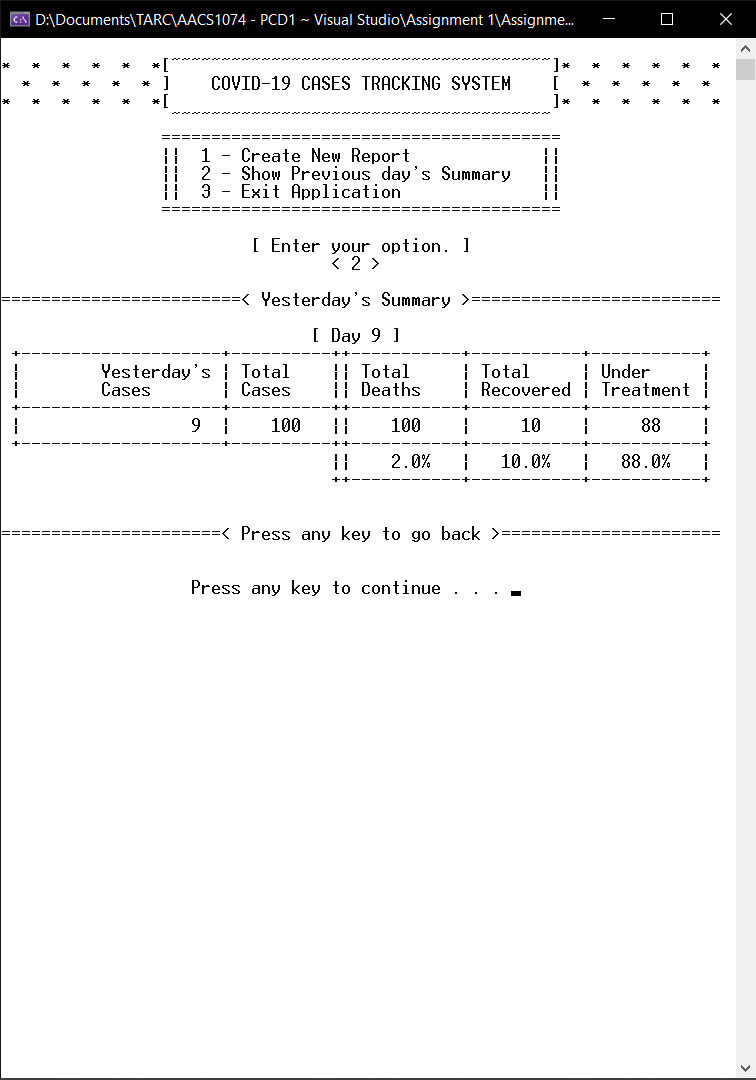
**6.4 Presentation of Extra Features**

**6.4.1 Title Screen**

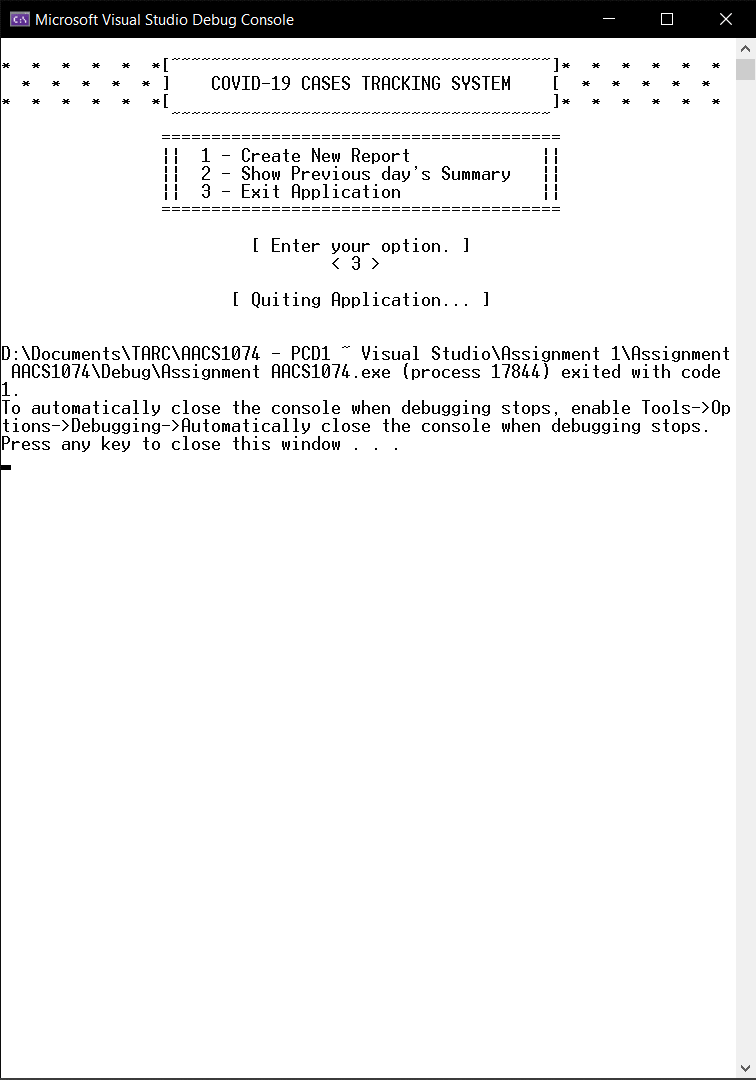
When the user chooses “1 - Create New Report”, the user will be able to start inputting data.

****

When the user chooses “2 – Show Previous day’s Summary”, it will display a previous day’s summary report.

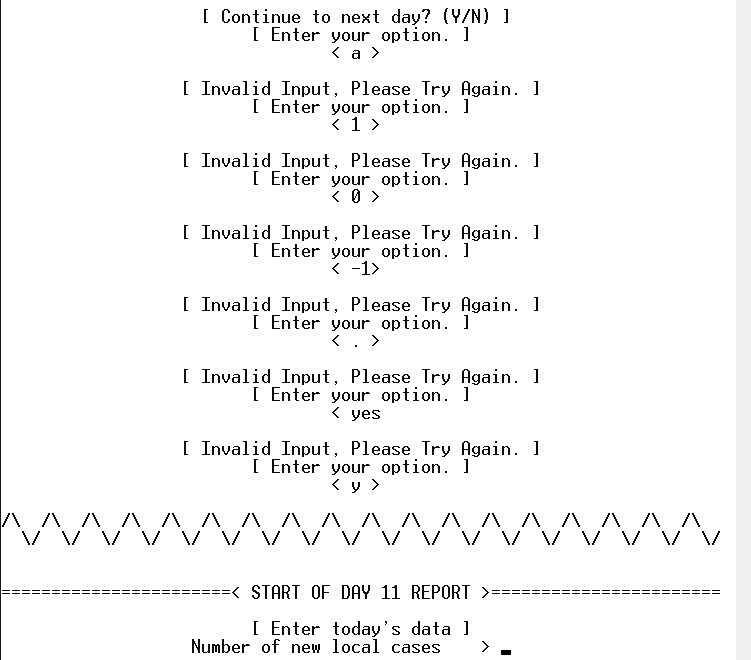
****

When the user choose “3 – Exit Application”, the program will quit.

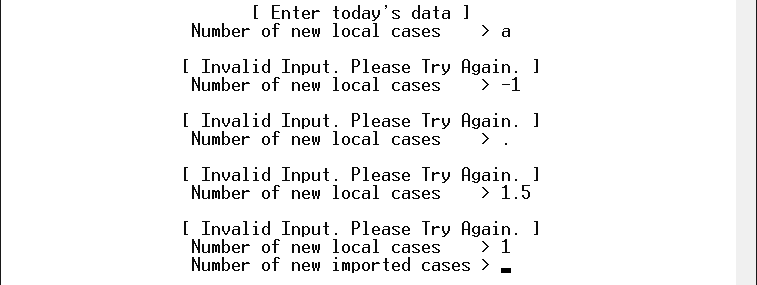


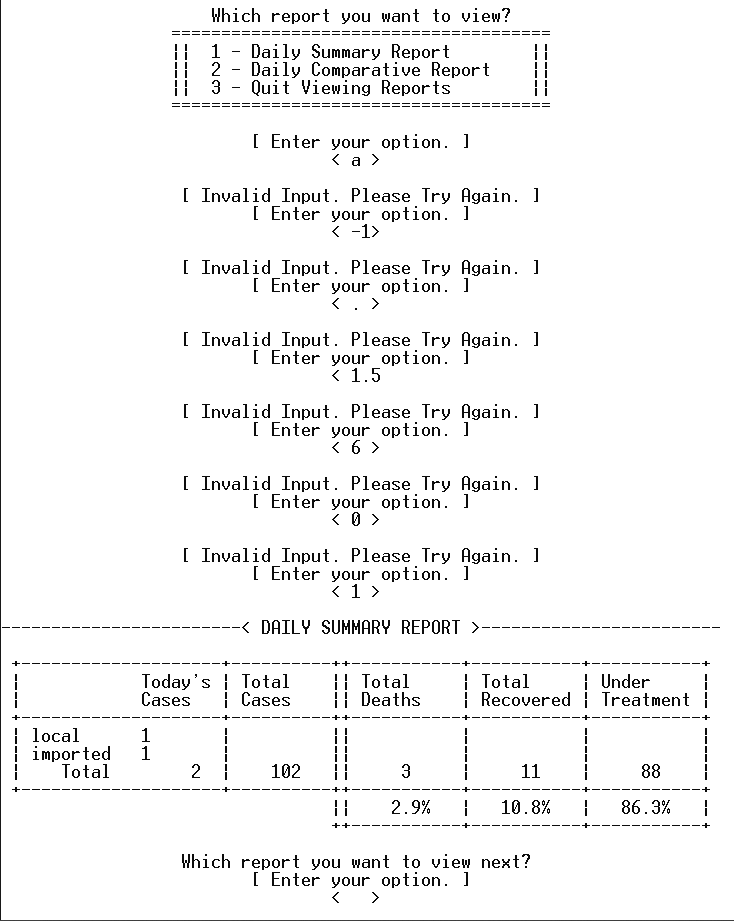
**6.4.2 Input Validation**

Validation for characters, only ‘y’, ‘Y’, ‘n’, ‘N’, ‘x’ and ‘X’ are allowed.

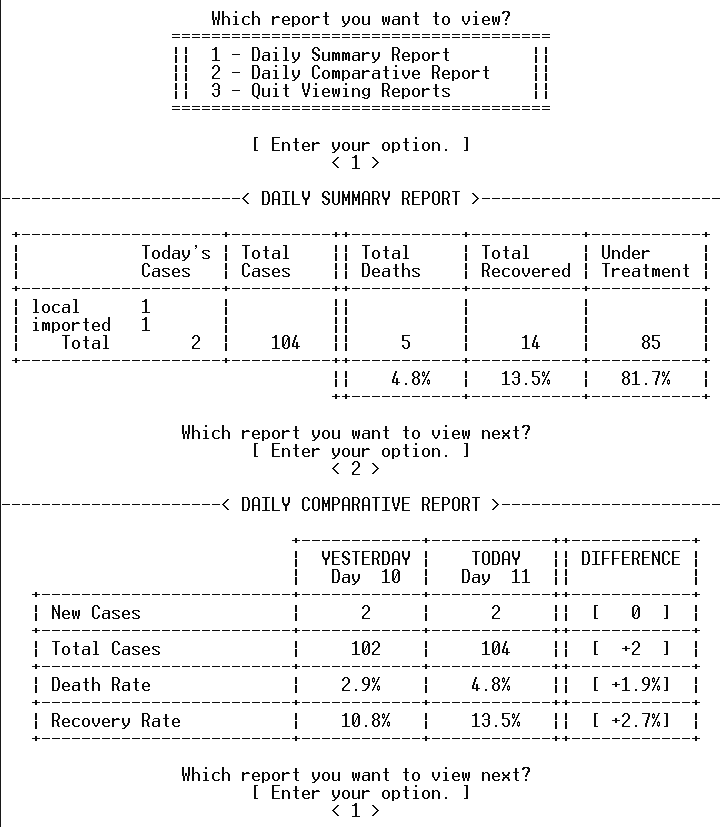
****

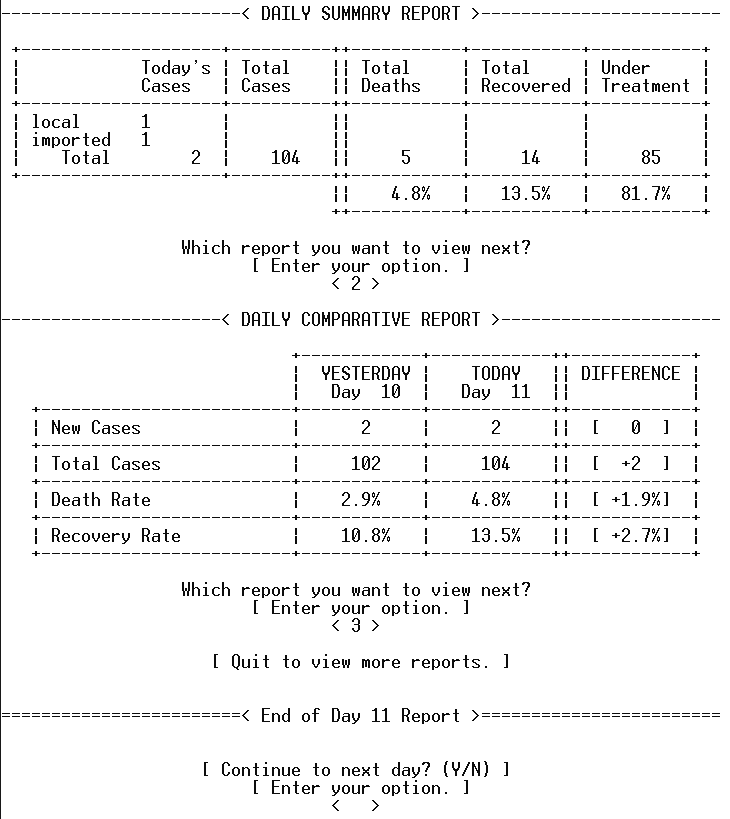
Validation for integers, only positive integers including zero are accepted.



Validation for integers at menus, only the option numbers stated in the menu are allowed.

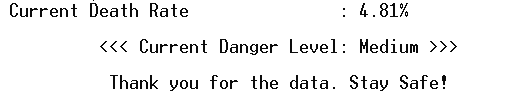
**6.4.3 View Reports Repeatedly**

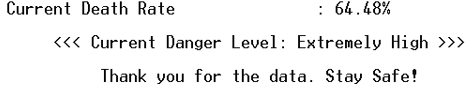
****Allows user to print reports repeatedly and indefinitely until user chooses “3 – Quit Viewing Reports”.

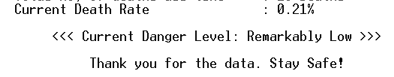
****

**6.4.4 Danger Level Section**

Displays current danger level at the ending section depending on the latest death rate.

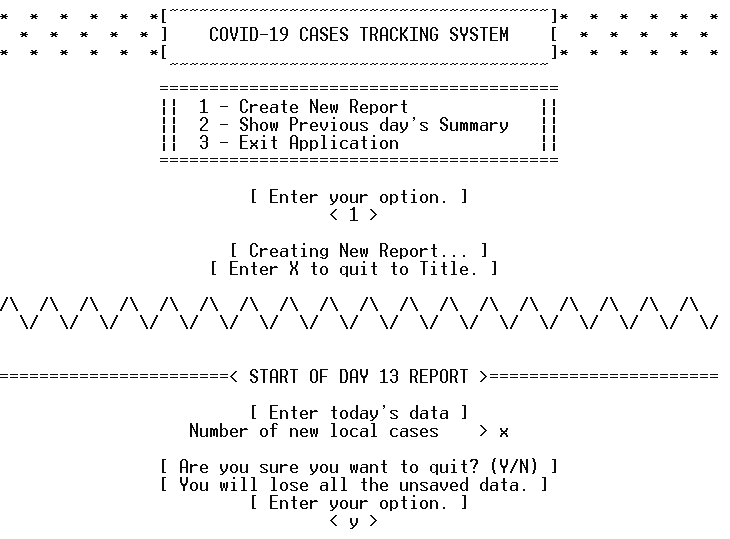
****

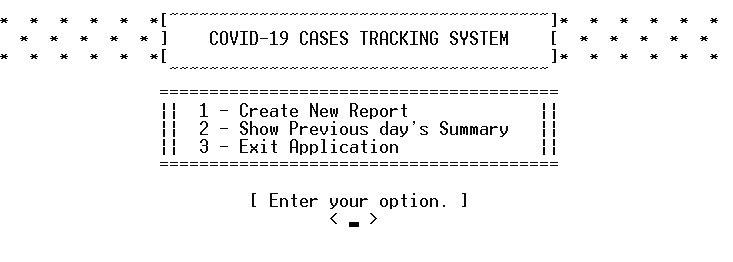
****

****

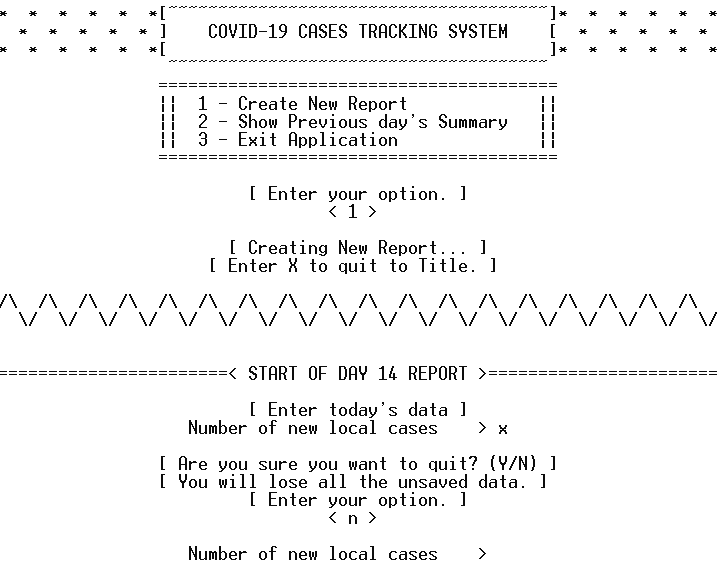
**6.4.5 Quit to Title Screen**

When the user enters ‘x’ or ‘X’, a double confirmation will be prompt before quitting to title. If enters ‘y’ or ‘Y’, the program will clear everything and quit to the title screen.

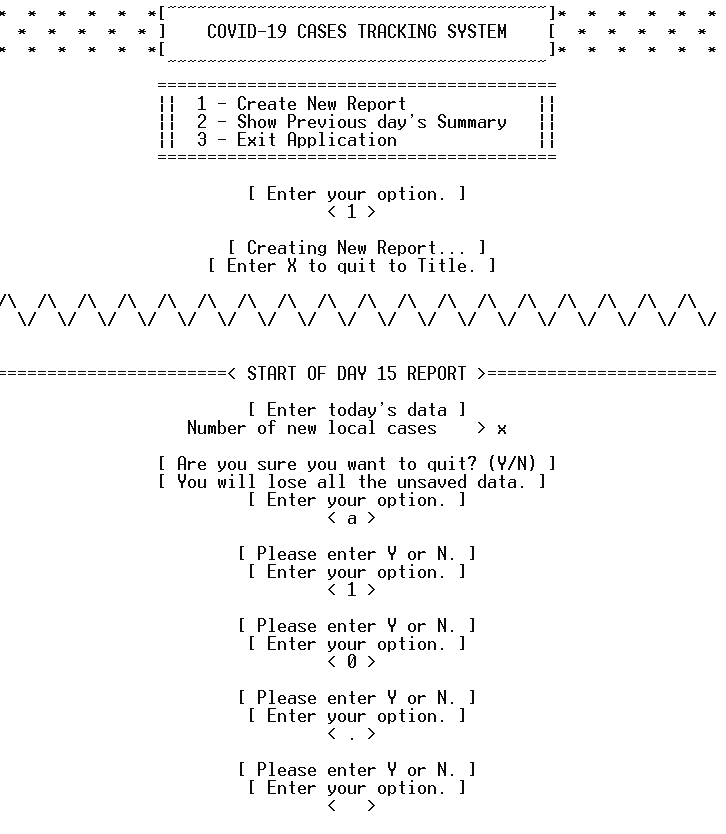
****

****

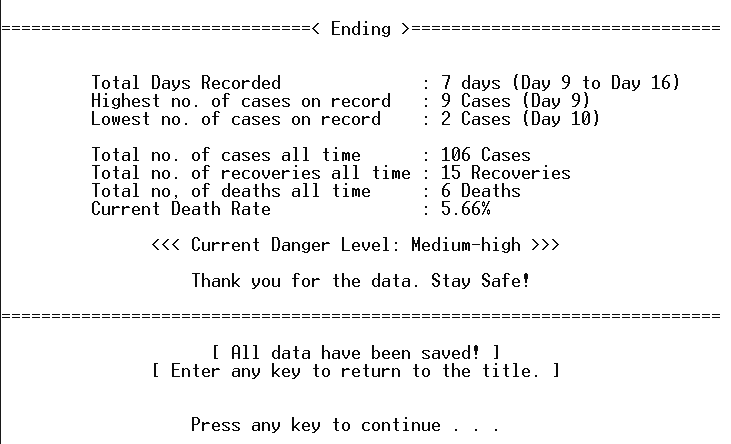
If the user enters ‘n’ or ‘N’ at the double confirmation, the user will be brought back to where it enters ‘x’ or ‘X’.

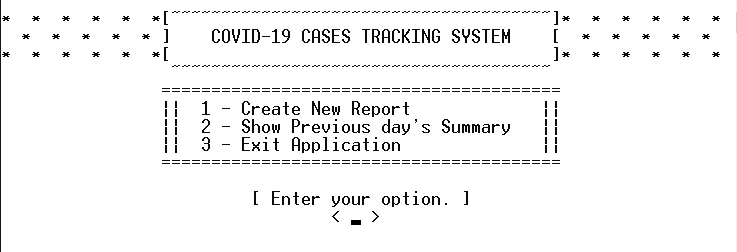
****

The double confirmation input field only allows ‘y’, ‘Y’, ‘n’ or ‘ N’ to be entered.

****

The user will be also brought back to the title screen after the ending screen.

****

****

**7.0 Appendix / Program Listing**

#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

#include <string.h>

#pragma warning(disable:4996)

/\*==================== DECLARE USER-DEFINED FUNCTIONS ====================\*/

// Main Functions

void centerAlign(char str[]); // Center aligns any given string, but refuses escape characters such as '\n'

void optionScreen(void); // [Input] Display the layout to get option input from user

int inputInt(int atTitle); // [Input] Get and validate integer input

char inputAlpha(int atTitle); // [Input] Get and validate character input (Y/N)

int returnTitle(int atTitle); // [Input] Double confirm if user wants to quit to the title

void reportMenu(int viewCount); // [Report] Ask to print which report

void dangerLevel(float deathRate); // [Ending] Danger level according to death rate

// Title Screen Functions

void programTitle(void); // Instantly prints Program Title at the title screen

void menuScreen(void); // Instantly prints Menu Screen at the title screen

void titleAnimation(void); // Title scroll in animation on startup

void menuAnimation(void); // Menu dropdown animation on startup

void printReverse(char line[], int def, int i); // Scroll in Animation for each line

void waitTimer(int timer); // A timer that waits for a given time

// Report Functions

void centerAlignInt(int data, int width, int haveBrackets); // Center align integers in a given width

void centerAlignFloat(float data, int width, int haveBrackets); // Center align floats in a given width

void centerAlignData(char str[], int width, int haveBrackets); // Main center align function for both functions above

void pdrHeader(void); // Prints Header of Previous Day's Report

void dsrHeader(void); // Prints Header of Daily Summary Report

void dcrHeader(void); // Prints Header of Daily Comparative Report

void dcrLine(void); // Prints Line Breaks of Daily Comparative Report

// Other Functions

void dayBreak(void); // Prints line breaks between days

void fill(int count, char ch); // Prints a given amount of the given character

/\*==================== CONSTANTS ====================\*/

// Initial Day's Data

const int INI\_NEW\_CASES = 9;

const int INI\_TOTAL\_CASES = 100;

const int INI\_DEATHS = 2;

const int INI\_RECOVERED = 10;

const int INI\_DAY = 9;

// Page width fixed to 72

const int PAGE\_WIDTH = 72;

// Line breaks

const char LINE1[] = "--------------------------------------------------";

const char LINE2[] = "==================================================";

/\*==================== MAIN FUNCTION ====================\*/

int main() {

/\*==================== VARIABLES ====================\*/

// Data variables

int preNew, preTotal, preDeaths, preRecovered, preTreatment; // Previous day's data

float preDeathRate, preRecoveredRate, preTreatmentRate; // Previous day's rates

int newLocals, newImported, newTotal, newDeaths, newRecovered; // Today's data

int totalCases, totalDeaths, totalRecovered, totalTreatment; // Total of (All Previous + Today) data

float deathRate, recoveredRate, treatmentRate; // Today's rates

int newChanges; // Today's new cases - Previous day's new cases

int currentDay, daycount; // Day variables

int highest, lowest, highestDay, lowestDay; // The highest and lowest cases and their respective days

// Option selections

int menuSelect, reportSelect; // Get option selections for the title menu and the report menu

char contiLoopDay; // Get Option selection for "Continue to next day? (Y/N)"

// Report variables

int viewCount; // Counts how many times the user had viewed the report

// Title screen variables

int atTitle = 0; // Whether the user is at the title screen.

/\*==================== DATA INITIALIZATION ====================\*/

/\* Previous day's data = Initial day's data \*/

currentDay = INI\_DAY;

preNew = INI\_NEW\_CASES;

preTotal = INI\_TOTAL\_CASES;

preDeaths = INI\_DEATHS;

preRecovered = INI\_RECOVERED;

preTreatment = preTotal - preDeaths - preRecovered;

/\* Initialize Highest and lowest value \*/

highest = INI\_NEW\_CASES;

lowest = INI\_NEW\_CASES;

highestDay = currentDay;

lowestDay = currentDay;

/\* Calculations for Initial day's data \*/

preDeathRate = (float)preDeaths / preTotal \* 100;

preRecoveredRate = (float)preRecovered / preTotal \* 100;

preTreatmentRate = (float)preTreatment / preTotal \* 100;

/\*==================== INTRO ANIMATION ====================\*/

// Title Logo

titleAnimation();

system("cls");

programTitle();

// Menu Options

menuAnimation();

system("cls");

// User who enters 'x' or reaches the ending screen will be brought back to here

// THE ONLY WAY to quit the program is by selecting "3 - Quit Application" at the title screen

///\* RETURN TO TITLE LOOP STARTS HERE \*///

while(1) { // Loop is always true

/\*================================= MENU SCREEN ========================================\*/

// Prints the Logo and Menu screen Instantly (Primes the menu loop below)

programTitle();

menuScreen();

// Prints when user enters 'x' at title screen

if (atTitle) {

centerAlign("[ Already at Title Screen. ]");

printf("\n");

}

// Indicate the user is at the title screen

atTitle = 1;

///\* TITLE SCREEN LOOP STARTS HERE \*///

do {

// Prompt and get menu option selection from user

do {

optionScreen();

menuSelect = inputInt(atTitle); // Get and validate input

if (menuSelect == -999) {

system("cls");

programTitle();

menuScreen();

centerAlign("[ Invalid Input. Please Try Again. ]");

printf("\n");

}

// If user enters 'x'

if (menuSelect == -69)

goto mainTitle; // Quit to title if choose 'y'

} while (menuSelect < 0); // Ask again if user enters invalid input / if user enters 'n' after entering 'x'

switch (menuSelect)

{

/\* 1 - Start inputting data \*/

case 1:

printf("\n");

centerAlign("[ Creating New Report... ]");

printf("\n");

centerAlign("[ Enter X to quit to Title. ]");

printf("\n");

break;

/\* 2 - Display Previous day's Report \*/

case 2:

// Break Line

printf("\n%.24s< Yesterday's Summary >%.25s\n\n", LINE2, LINE2);

centerAlign("[ Day ]");

fill(3, '\b');

printf("%d ]\n", currentDay);

// Header

printf(" +%.20s+%.10s++%.11s+%.11s+%.11s+\n", LINE1, LINE1, LINE1, LINE1, LINE1);

printf(" |%20s|%-10s||%-11s|%-11s|%-11s|\n", "Yesterday's ", " Total", " Total", " Total", " Under");

printf(" |%20s|%-10s||%-11s|%-11s|%-11s|\n", "Cases ", " Cases", " Deaths", " Recovered", " Treatment");

// Main Data

printf(" +%.20s+%.10s++%.11s+%.11s+%.11s+\n", LINE1, LINE1, LINE1, LINE1, LINE1);

printf(" |%18d |", preNew);

centerAlignInt(preTotal, 10, 0);

printf("|");

centerAlignInt(preTotal, 11, 0);

centerAlignInt(preRecovered, 11, 0);

centerAlignInt(preTreatment, 11, 0);

printf("\n");

// Rates

printf(" +%.20s+%.10s++%.11s+%.11s+%.11s+\n", LINE1, LINE1, LINE1, LINE1, LINE1);

printf(" %32s||", "");

centerAlignFloat(preDeathRate, 11, 0);

centerAlignFloat(preRecoveredRate, 11, 0);

centerAlignFloat(preTreatmentRate, 11, 0);

printf("\n");

printf(" %32s++%.11s+%.11s+%.11s+\n\n", "", LINE1, LINE1, LINE1);

//Ending Line

printf("\n%.22s< Press any key to go back >%.22s\n\n\n", LINE2, LINE2);

centerAlign("Press any key to continue . . .");

fill(32, '\b');

system("pause");

// Return to title screen

system("cls");

programTitle();

menuScreen();

break;

/\* 3 - Exit the program \*/

case 3:

printf("\n");

centerAlign("[ Quiting Application... ]");

printf("\n\n");

exit(1);

break;

/\* Invalid Input (Integer not in options ) \*/

default:

system("cls");

programTitle();

menuScreen();

centerAlign("[ Invalid Input. Please Try Again. ]");

printf("\n");

break;

}

} while (menuSelect != 1);

///\* TITLE SCREEN LOOP ENDS HERE \*///

// User is no longer at the title;

atTitle = 0;

///\* DAY LOOP STARTS HERE \*///

do {

// New Day

currentDay++;

dayBreak();

printf("\n\n%.23s< START OF DAY %d REPORT >%.23s\n\n", LINE2, currentDay, LINE2);

/\*================================== Input Section =======================================\*/

// Input: New local cases

centerAlign("[ Enter today's data ]");

printf("\n");

do {

centerAlign("Number of new local cases > ");

fill(3, '\b');

newLocals = inputInt(atTitle);

// If user enters 'x' and chooses 'y'

if (newLocals == -69)

goto mainTitle;

} while (newLocals < 0); // Ask again if input is invalid or entered 'n' at quit to title double confirmation

// Input: New Imported cases

do {

centerAlign("Number of new imported cases > ");

fill(3, '\b');

newImported = inputInt(atTitle);

// If user enters 'x' and chooses 'y'

if (newImported == -69)

goto mainTitle;

} while (newImported < 0); // Ask again if input is invalid or entered 'n' at quit to title double confirmation

// Short Summary of First two Input

newTotal = newLocals + newImported; // Total new cases

newChanges = newTotal - preNew; // Changes

// Print changes

printf("\n");

centerAlign("[ Total number of new cases = ]");

fill(4, '\b');

printf("%d ] \n", newTotal);

if (newChanges == 0) {

centerAlign("( Same as yesterday. )", ' ');

}

else {

if (newChanges > 0) {

centerAlign("( Increased by xx from yesterday. )");

}

else {

centerAlign("( Decreased by xx from yesterday. )");

}

fill(20, '\b');

printf("%d from yesterday. ) ", newChanges);

}

printf("\n\n");

// Input: New Deaths

do {

centerAlign("Number of new deaths > ");

fill(3, '\b');

newDeaths = inputInt(atTitle);

// If user enters 'x' and chooses 'y'

if (newDeaths == -69)

goto mainTitle;

} while (newDeaths < 0); // Ask again if input is invalid or entered 'n' at quit to title double confirmation

// Input: New recoveries

do {

centerAlign("Number of new recoveries > ");

fill(3, '\b');

newRecovered = inputInt(atTitle);

// If user enters 'x' and chooses 'y'

if (newRecovered == -69)

goto mainTitle;

} while (newRecovered < 0); // Ask again if input is invalid or entered 'n' at quit to title double confirmation

printf("\n");

/\*================================== Calculations =======================================\*/

// Main data calculation

totalCases = preTotal + newTotal;

totalDeaths = preDeaths + newDeaths;

totalRecovered = preRecovered + newRecovered;

totalTreatment = totalCases - totalDeaths - totalRecovered;

// Rates calculation

deathRate = (float)totalDeaths / totalCases \* 100;

recoveredRate = (float)totalRecovered / totalCases \* 100;

treatmentRate = (float)totalTreatment / totalCases \* 100;

// Check for highest cases

if (newTotal > highest) {

highest = newTotal;

highestDay = currentDay;

}

// Check for lowest cases

if (newTotal < lowest) {

lowest = newTotal;

lowestDay = currentDay;

}

/\*=================================== Printing Report ======================================\*/

viewCount = 1; // Reset view

reportMenu(viewCount); // Show the report options

///\* REPORT LOOP STARTS HERE \*///

do {

// Ask to print which report

optionScreen(); // Ask for input

reportSelect = inputInt(atTitle); // Get and validate input

// If user enters 'x' and choose 'y'

if (reportSelect == -69)

goto mainTitle;

// If user enters 'x' and choose 'n'

else if (reportSelect == -70)

reportMenu(viewCount);

/\* 1 - Displays "Daily Summary Report" --> Re-enter Input \*/

else if (reportSelect == 1) {

// ================================ DSR Table Visualized ================================

// +------------------+-------------++--------------+-----------------+-----------------+

// | Today's cases | Total cases || Total Deaths | Total Recovered | Under Treatment |

// +------------------+-------------++--------------+-----------------+-----------------+

// | local (D1) | || : : |

// | imported (D2) | || : : |

// | (Total) | ( D3 ) || : ( D4 ) : |

// +------------------+-------------++--------------+-----------------+-----------------+

// || : ( RATES ) : |

// ++--------------+-----------------+-----------------+

// Header

dsrHeader();

// Main Data

printf(" | local %-8d|%10s||%11s|%11s|%11s|\n", newLocals, "", "", "", ""); // (D2)

printf(" | imported %-8d|%10s||%11s|%11s|%11s|\n", newImported, "", "", "", ""); // (D1)

printf(" | Total %6d |", newTotal); // Total

// (D3)

centerAlignInt(totalCases, 10, 0);

printf("|");

// (D4)

centerAlignInt(totalDeaths, 11, 0);

centerAlignInt(totalRecovered, 11, 0);

centerAlignInt(totalTreatment, 11, 0);

printf("\n");

// (RATES)

printf(" +%.20s+%.10s++%.11s+%.11s+%.11s+\n", LINE1, LINE1, LINE1, LINE1, LINE1);

printf(" %32s||", "");

centerAlignFloat(deathRate, 11, 0);

centerAlignFloat(recoveredRate, 11, 0);

centerAlignFloat(treatmentRate, 11, 0);

printf("\n %32s++%.11s+%.11s+%.11s+\n\n", "", LINE1, LINE1, LINE1);

// Prints the menu for the next loop

viewCount++;

reportMenu(viewCount);

}

/\* 2 - Displays "Daily Comparative Report" --> Re-enter input \*/

else if (reportSelect == 2) {

// ================== DCR Table Visualized ==================

// +-----------+-----------++---------------+

// | Yesterday | Today || Difference |

// +---------------+-----------+-----------++---------------+

// | New Cases | C1R1 | C2R1 || [+- C3R1 ] |

// +---------------+-----------+-----------++---------------+

// | Total Cases | C1R2 | C2R2 || [+- C3R2 ] |

// +---------------+-----------+-----------++---------------+

// | Death Rate | C1R3 % | C2R3 % || [+- C3R3 %] |

// +---------------+-----------+-----------++---------------+

// | Recovery Rate | C1R4 % | C2R4 % || [+- C3R4 %] |

// +---------------+-----------+-----------++---------------+

// Header

dcrHeader();

printf(" %26s| Day%4d | Day%4d ||%12s|\n", "", currentDay - 1, currentDay, "");

dcrLine();

// Row 'New Cases'

printf(" | New Cases%15s|", "");

centerAlignInt(preNew, 12, 0); // (C1R1)

centerAlignInt(newTotal, 12, 0); // (C2R1)

printf("|"); // '|'

centerAlignInt(newChanges, 12, 1); // (C3R1)

dcrLine(); // ----Line Break----

// Row 'Total Cases'

printf(" | Total Cases%13s|", "");

centerAlignInt(preTotal, 12, 0); // (C1R2)

centerAlignInt(totalCases, 12, 0); // (C2R2)

printf("|"); // '|'

centerAlignInt(totalCases - preTotal, 12, 1); // (C2R3)

dcrLine(); // ----Line Break----

// Row 'Death Rate'

printf(" | Death Rate%14s|", "");

centerAlignFloat(preDeathRate, 12, 0); // (C1R3)

centerAlignFloat(deathRate, 12, 0); // (C2R3)

printf("|"); // '|'

centerAlignFloat(deathRate - preDeathRate, 12, 1); // (C3R3)

dcrLine(); // ----Line Break----

// Row 'Recovery Rate'

printf(" | Recovery Rate%11s|", "");

centerAlignFloat(preRecoveredRate, 12, 0); // (C1R4)

centerAlignFloat(recoveredRate, 12, 0); // (C2R4)

printf("|"); // '|'

centerAlignFloat(recoveredRate - preRecoveredRate, 12, 1); // (C3R4)

dcrLine(); // ----Line Break----

printf("\n");

// Prints the menu for the next loop

viewCount++;

reportMenu(viewCount);

}

/\* 3 - Quit viewing reports \*/

else if (reportSelect == 3) {

printf("\n");

if (viewCount == 1)

centerAlign("[ Quit to view any reports. ]");

else

centerAlign("[ Quit to view more reports. ]");

}

/\* Wrong Input --> Re-enter Input \*/

else if (reportSelect != -999 && reportSelect != -70) {

printf("\n");

centerAlign("[ Invalid Input. Please Try Again. ]");

printf("\n");

}

} while (reportSelect != 3); // Always repeat unless user choose quit

///\* REPORT LOOP ENDS HERE \*///

// End of day

printf("\n\n\n%.24s< End of Day %d Report >%.24s\n\n\n", LINE2, currentDay, LINE2);

/\*================================ Ask to Continue =========================================\*/

// Ask for continuation

centerAlign("[ Continue to next day? (Y/N) ]");

printf("\n");

do {

// Get user option selection

optionScreen();

contiLoopDay = inputAlpha(atTitle);

// If user enters 'x'

if (contiLoopDay == 'x')

goto mainTitle;

} while (contiLoopDay == 'z'); // 'z' == Invalid input

/\* Only 'y' and 'n' are allowed to exit loop \*/

/\* Today's data --> Previous day's data \*/

preNew = newTotal;

preTotal = totalCases;

preDeaths = totalDeaths;

preRecovered = totalRecovered;

preTreatment = totalTreatment;

preDeathRate = deathRate;

preRecoveredRate = recoveredRate;

preTreatmentRate = treatmentRate;

// 'Y' = Continue to next day; 'N' = Go to Ending

} while (contiLoopDay == 'y');

///\* DAY LOOP ENDS HERE \*///

/\*================================ Ending =========================================\*/

// Line Break

dayBreak();

printf("\n\n%.31s< Ending >%.31s\n\n\n", LINE2, LINE2);

// Total days recorded (singular or plural)

if (currentDay - INI\_DAY == 1)

printf("\t Total Days Recorded : %d day (Day %d to Day %d)\n", currentDay - INI\_DAY, INI\_DAY, currentDay);

else

printf("\t Total Days Recorded : %d days (Day %d to Day %d)\n", currentDay - INI\_DAY, INI\_DAY, currentDay);

printf("\t Highest no. of cases on record : %d Cases (Day %d)\n", highest, highestDay);

printf("\t Lowest no. of cases on record : %d Cases (Day %d)\n\n", lowest, lowestDay);

// Statistics

printf("\t Total no. of cases all time : %d Cases\n", totalCases);

printf("\t Total no. of recoveries all time : %d Recoveries\n", totalRecovered);

printf("\t Total no, of deaths all time : %d Deaths\n", totalDeaths);

printf("\t Current Death Rate : %.2f%%\n\n", deathRate);

// Danger Level

dangerLevel(deathRate);

// Pleasant Ending

printf("\n\n");

centerAlign("Thank you for the data. Stay Safe!");

printf("\n\n");

printf("%.50s%.22s\n\n", LINE2, LINE2);

// Tell user the data inputted are saved and will be redirected back to title

centerAlign("[ All data have been saved! ]");

printf("\n");

centerAlign("[ Enter any key to return to the title. ]");

printf("\n\n\n");

centerAlign("Press any key to continue . . .");

fill(32, '\b');

system("pause");

// User who enters 'x' at any input and 'y' at double confirmation will skip everything and jump to this postition

mainTitle:

system("cls");

}

///\* RETURN TO TITLE LOOP ENDS HERE \*///

// The only way to quit the program is via "3 - Exit Application" in the title screen

/\*=============================== End of Program ==========================================\*/

}

/\*================================ User-Defined Functions =========================================\*/

/\*=============== MAIN FUNCTIONS ===============\*/

// Center aligns any given string, but refuses escape characters such as '\n'

void centerAlign(char str[]) {

int length = strlen(str);

int indent = (PAGE\_WIDTH - length) / 2;

fill(indent, ' ');

printf("%s", str);

}

// [Input] Display the layout to get option input from user

void optionScreen(void) {

centerAlign("[ Enter your option. ]");

printf("\n");

centerAlign("< >");

fill(3, '\b');

}

// [Input] Get and validate integer input

int inputInt(int atTitle) {

char input[13];

int i; // Empty

int quitToTitle;

/\* RETURN TYPES

\*\* input = Valid input, Range(0, infinite)

\*\* -999 = Invalid Input - Input too long / Contain not integers

\*\* -69 = Entered X --> Double confirm --> Input 'y'

\*\* -70 = Entered X -- > Double confirm --> Input 'n'

\*\* -420 = Out of Range (When inputting option selections)

\*/

// Get input in string form

rewind(stdin);

scanf("%s", input);

// If user enters 'x'

if (strlen(input) == 1 && tolower(input[0]) == 'x') {

quitToTitle = returnTitle(atTitle);

if (quitToTitle)

return -69;

else

return -70;

}

// Validate if the input is an positive integer

for (i = 0; i < strlen(input); i++) {

if ((!(isdigit(input[i])))) {

printf("\n");

if (!atTitle) {

centerAlign("[ Invalid Input. Please Try Again. ]");

}

printf("\n");

return -999; // Indicates invalid input

}

}

// If everything is ok,

// Convert input string to integer and return

return atoi(input);

}

// [Input] Get and validate character input

char inputAlpha(int atTitle) {

char input[13], invalid = 'z';

int quitToTitle;

/\* RETURN TYPES

\*\* 'y' = Choosed Yes

\*\* 'n' = Choosed No

\*\* 'x' = Entered X --> Double confirm --> Quit to Title

\*\* 'z' = Invalid Input, when entered anything else then y, n or x

\*/

// Get input in string form

rewind(stdin);

scanf("%s", input);

// Validate and only allow 'y', 'n' or 'x' to be inputted

if (strlen(input) == 1) {

if (tolower(input[0]) == 'y')

return 'y';

else if (tolower(input[0]) == 'n')

return 'n';

else if (tolower(input[0]) == 'x') {

quitToTitle = returnTitle(atTitle);

if (quitToTitle)

return 'x';

else {

// Asked again if user entered invalid input

centerAlign("[ Continue to next day? (Y/N) ]");

printf("\n");

return 'z';

}

}

}

// User will be prompted again if entered invalid input

printf("\n");

centerAlign("[ Invalid Input, Please Try Again. ]");

printf("\n");

return 'z';

}

// [Input] Double confirm if user wants to quit to the title

int returnTitle(int atTitle) {

char option, input[13];

int valid = 0;

// Clear the screen if user is already at the title

if (atTitle) {

system("cls");

return 1;

}

// Ask for double confirmation if user really wants to quit to the title screen

printf("\n");

centerAlign("[ Are you sure you want to quit? (Y/N) ]");

printf("\n");

centerAlign("[ You will lose all the unsaved data. ]");

printf("\n");

optionScreen();

rewind(stdin);

scanf("%s", input);

// If user enters 'y' or 'n'

if (strlen(input) == 1 && (tolower(input[0]) == 'y' || tolower(input[0]) == 'n')) {

valid = 1;

}

// Ask again if user entered invalid input

while (!valid) {

printf("\n");

centerAlign("[ Please enter Y or N. ]");

printf("\n");

optionScreen();

rewind(stdin);

scanf("%s", input);

// Only exit the loop if user had entered 'y' or 'n'

if (strlen(input) == 1 && (tolower(input[0]) == 'y' || tolower(input[0]) == 'n')) {

valid = 1;

}

}

printf("\n");

// 1 = Quit to title; 0 = Don't quit to title;

if (tolower(input[0]) == 'y')

return 1;

else

return 0;

}

// [Report] Ask to print which report

void reportMenu(int viewCount) {

if (viewCount == 1) {

printf("\n");

centerAlign("Which report you want to view?");

printf("\n");

centerAlign("======================================");

printf("\n");

centerAlign("|| 1 - Daily Summary Report ||");

printf("\n");

centerAlign("|| 2 - Daily Comparative Report ||");

printf("\n");

centerAlign("|| 3 - Quit Viewing Reports ||");

printf("\n");

centerAlign("======================================");

printf("\n\n");

}

else {

centerAlign("Which report you want to view next?");

printf("\n");

}

}

// [Ending] Displays danger level according to Death Rate

void dangerLevel(float deathRate) {

char dangerLevel[20], output[50] = "<<< Current Danger Level: ";

if (deathRate < 0.1)

strcpy(dangerLevel, "Extremely Low");

else if (deathRate < 0.25)

strcpy(dangerLevel, "Remarkably Low");

else if (deathRate < 0.5)

strcpy(dangerLevel, "Very Low");

else if (deathRate < 1)

strcpy(dangerLevel, "Low");

else if (deathRate < 3)

strcpy(dangerLevel, "Low-medium");

else if (deathRate < 5)

strcpy(dangerLevel, "Medium");

else if (deathRate < 7)

strcpy(dangerLevel, "Medium-high");

else if (deathRate < 10)

strcpy(dangerLevel, "High");

else if (deathRate < 30)

strcpy(dangerLevel, "Very High");

else if (deathRate < 50)

strcpy(dangerLevel, "Remarkably High");

else if (deathRate < 75)

strcpy(dangerLevel, "Extremely High");

else

strcpy(dangerLevel, "God Save The Earth");

strcat(output, dangerLevel);

strcat(output, " >>>");

centerAlign(output);

}

/\*=============== TITLE SCREEN FUNCTIONS ===============\*/

// Instantly prints program title at the title screen

void programTitle(void) {

printf("\n");

printf("\* \* \* \* \* \*[~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~]\* \* \* \* \* \*\n");

printf(" \* \* \* \* \* ] COVID-19 CASES TRACKING SYSTEM [ \* \* \* \* \* \n");

printf("\* \* \* \* \* \*[ ]\* \* \* \* \* \*\n");

printf(" ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ \n");

}

// Instantly prints menu screen at the title screen

void menuScreen(void) {

printf("\t\t========================================\n");

printf("\t\t|| 1 - Create New Report ||\n");

printf("\t\t|| 2 - Show Previous day's Summary ||\n");

printf("\t\t|| 3 - Exit Application ||\n");

printf("\t\t========================================\n\n");

}

// Title scroll in animation on startup

void titleAnimation(void)

{

// Title string

char line1[] = "\* \* \* \* \* \*[~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~]\* \* \* \* \* \*";

char line2[] = " \* \* \* \* \* ] COVID-19 CASES TRACKING SYSTEM [ \* \* \* \* \* ";

char line3[] = "\* \* \* \* \* \*[ ]\* \* \* \* \* \*";

char line4[] = " ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ ";

int length = strlen(line1); // Length of title

int i = length; // Position to start printing title

int blinkStart = 3; // Time before start blinking

int blinkInterval = 2; // Time intervals when blinking

int blinkEnd = 8; // Time after end blinking

while (i != -1) {

// Title scroll in from right

printf("\n");

printReverse(line1, length, i);

printReverse(line2, length, i);

printReverse(line3, length, i);

printReverse(line4, length, i);

// Title blinking animation

if (i == 0) {

waitTimer(blinkStart);

// Blink 3 times

for (int n = 1; n <= 3; n++) {

system("cls");

printf("\n\n\n\n\n");

waitTimer(blinkInterval);

system("cls");

printf("\n%s\n%s\n%s\n%s\n", line1, line2, line3, line4);

waitTimer(blinkInterval);

}

waitTimer(blinkEnd);

break;

}

// Clear screen for this frame

system("cls");

i--;

}

}

// Menu dropdown animation on startup

void menuAnimation(void)

{

// Menu options string

char line1[] = "========================================";

char line2[] = "|| 1 - Create New Report ||";

char line3[] = "|| 2 - Show Previous day's Summary ||";

char line4[] = "|| 3 - Exit Application ||";

char line5[] = "========================================";

int length = strlen(line1); // Length of menu string

int timerInterval = 3; // Time interval between animation

int i; // Empty variable

// Line 1

printf("\t\t%s", line1);

waitTimer(timerInterval);

// Line 1, 5

printf("\n\t\t%s", line5);

waitTimer(timerInterval);

// Line 1, 2, 5

fill(length, '\b');

printf("%s", line2);

printf("\n\t\t%s", line5);

waitTimer(timerInterval);

// Line 1, 2, 3, 5

fill(length, '\b');

printf("%s", line3);

printf("\n\t\t%s", line5);

waitTimer(timerInterval);

// Line 1, 2, 3, 4, 5

fill(length, '\b');

printf("%s", line4);

printf("\n\t\t%s", line5);

waitTimer(timerInterval);

}

// Scroll in Animation for Each Line

void printReverse(char line[], int len, int i) {

// Print blank spaces

for (int n = 0; n < i; n++)

printf(" ");

// Print title

for (int n = 0; n <= len - i; n++)

printf("%c", line[n]);

printf("\n");

}

// A timer that waits for a given time

void waitTimer(int timer) {

timer \*= 100000000;

// Waste time by doing meaningless calculations

while (timer != 0)

timer--;

}

/\*=============== REPORT FUNCTIONS ===============\*/

// Center align integers in a given width

void centerAlignInt(int data, int width, int haveBrackets) {

char str[13];

// Convert data into string form

if (haveBrackets && data > 0)

sprintf(str, "+%d", data);

else

sprintf(str, "%d", data);

// Pass the string into center align function

centerAlignData(str, width, haveBrackets);

}

// Center align floats in a given width

void centerAlignFloat(float data, int width, int haveBrackets) {

char str[13];

int check = (data \* 10 + 0.5);

// Convert data into string form

if (haveBrackets && data > 0.0) // "+" prefix for positive floats

sprintf(str, "+%.1f%%", data);

else if (check == 0) // Prevent "+0.0" or "-0.0"

sprintf(str, "%.1f%%", abs(data));

else

sprintf(str, "%.1f%%", data);

// Pass the string into center align function

centerAlignData(str, width, haveBrackets);

}

// Main center align function for both functions above

void centerAlignData(char str[], int width, int haveBrackets) {

int length = strlen(str);

// For printing data without brackets

if (!haveBrackets) {

// Calculate left and right spaces

int rightSpaces = (width - length) / 2;

int leftSpaces = rightSpaces;

if ((width % 2 == 0 && length % 2 == 1) || (width % 2 == 1 && length % 2 == 0))

leftSpaces++;

// Print the data

fill(leftSpaces, ' ');

printf("%s", str);

fill(rightSpaces, ' ');

printf("|");

}

// For printing data with brackets

else {

// Calculate the left and right spaces

int outerSpace = 2;

if (strlen(str) > 8)

outerSpace = 0;

else if (strlen(str) > 6)

outerSpace = 1;

// The minimum width of bracket is set to 6

int innerWidth = width - 2 - (2 \* outerSpace);

int innerLeft = (innerWidth - length) / 2;

int innerRight = innerLeft;

if (length % 2 == 1)

innerLeft++;

// Print the data

fill(outerSpace, ' ');

printf("[");

fill(innerLeft, ' ');

printf("%s", str);

fill(innerRight, ' ');

printf("]");

fill(outerSpace, ' ');

printf("|\n");

}

}

// Prints Header of Previous Day's Report

void pdrHeader(void) {

printf(" +%.20s+%.10s++%.11s+%.11s+%.11s+\n", LINE1, LINE1, LINE1, LINE1, LINE1);

printf(" |%20s|%-10s||%-11s|%-11s|%-11s|\n", "Yesterday's ", " Total", " Total", " Total", " Under");

printf(" |%20s|%-10s||%-11s|%-11s|%-11s|\n", "Cases ", " Cases", " Deaths", " Recovered", " Treatment");

printf(" +%.20s+%.10s++%.11s+%.11s+%.11s+\n", LINE1, LINE1, LINE1, LINE1, LINE1);

}

// Prints Header of Daily Summary Report

void dsrHeader(void) {

printf("\n%.24s< DAILY SUMMARY REPORT >%.24s\n\n", LINE1, LINE1);

printf(" +%.20s+%.10s++%.11s+%.11s+%.11s+\n", LINE1, LINE1, LINE1, LINE1, LINE1);

printf(" |%20s|%-10s||%-11s|%-11s|%-11s|\n", "Today's ", " Total", " Total", " Total", " Under");

printf(" |%20s|%-10s||%-11s|%-11s|%-11s|\n", "Cases ", " Cases", " Deaths", " Recovered", " Treatment");

printf(" +%.20s+%.10s++%.11s+%.11s+%.11s+\n", LINE1, LINE1, LINE1, LINE1, LINE1);

}

// Prints Header of Daily Comparative Report

void dcrHeader(void) {

printf("\n%.22s< DAILY COMPARATIVE REPORT >%.22s\n\n", LINE1, LINE1);

printf(" %26s+%.12s+%.12s++%.12s+\n", "", LINE1, LINE1, LINE1);

printf(" %26s| YESTERDAY | TODAY || DIFFERENCE |\n", "");

}

// Prints Line Breaks of Daily Comparative Report

void dcrLine(void) {

printf(" +%.25s+%.12s+%.12s++%.12s+\n", LINE1, LINE1, LINE1, LINE1);

}

/\*=============== OTHER FUNCTIONS ===============\*/

// Prints line breaks between days

void dayBreak(void) {

int i;

printf("\n");

for (i = 1; i <= 18; i++) {

printf("/\\ ");

}

printf("\n");

for (i = 1; i <= 18; i++) {

printf(" \\/");

}

printf("\n");

}

// Prints the given amount of the given character

void fill(int count, char ch) {

int i;

for (i = 0; i < count; i++)

printf("%c", ch);

}