COMP1011 Group 10 Project Design Report

Project title: Introducing recursion

19082848D ZHANG Yujing
20076516D LIU Yang
20084222D ZHAO Zian
20077711D WANG Zehan

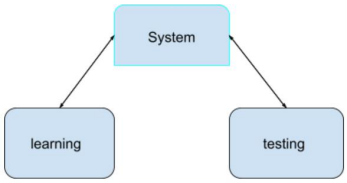
1. Overview

Introductory to recursion system is used to introduce recursion to the users. In this system, the user could understand what is recursion, how to apply it , and even use it to solve the real life problem step by step. In order to better fit the user's use needs, we design 2 mode in our system: learning mode and testing mode. In this report, we will show you how we design this system.

2. Design

2.1 Structure

The project includes 2 modes: learning mode and testing mode:



2.1.1 Learing Mode:

For the learning mode, we search for many materials about the recursion and then we input the learning source input different levels to make sure the difficulty is from the easy to the difficult and the knowledge is easy to understand. In the learning mode, there are totally 6 concepts :

```
1. What is recursion?
2. What is base condition in recursion?
3. How a particular problem is solved using recursion?
4. Concept differentiation: recursion and iteration
5. Why Stack Overflow error occurs in recursion?
6. What is Tail Recursion?
7. Back to the main directory.
```

These 6 parts cover the basic concept, special cases, concept discrimination and practical application. The user could have a comprehensive understanding of recursion after learning them.

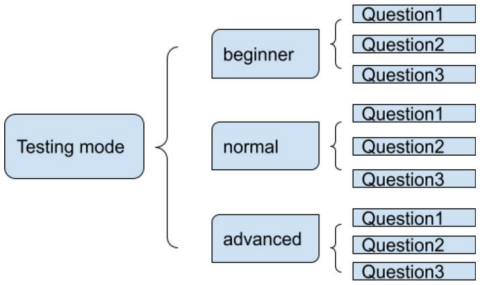
Besides, when introducing the related concepts, we also attach some example code to assist the user to better understand the concept of recursion.

```
#######################################
# What is base condition in recursion? #
#######################################

In the recursive program, the solution to the base case is provided and the
solution of the bigger problem is expressed in terms of smaller problems.

 +---------------------------------+
 |  int fact(int n){               |
 |     if (n < = 1) // base case   |
 |        return 1;               |
 |     else                        |
 |        return n*fact(n-1);      |
 +---------------------------------+

In the above example, base case for n < = 1 is defined and larger value of number
can be solved by converting to smaller one till base case is reached.
```

As above shows, when we introduce the base condition of recursion, a piece of code is included to show what is the base case.

2.1.2 Testing Mode

In the testing mode, we provide the users with differenty levels so that the user could choose the difficulty level based on their own mastery of the concept.



Following is a list of what is tested for each question:

Beginner:

1. Basic concept of recursion.
2. Simple application of recursion
3. The base condition of recursion

Normal:

1. How to control the number of recursion
2. The application of nested recursion
3. When the recursion should be used

Advanced:

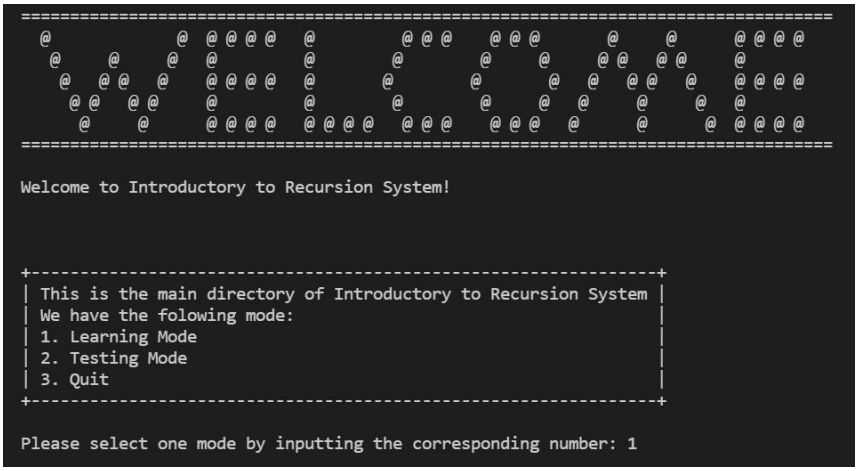
1. More complex recursive applicaTions (Eight Queens Question)
2. More complex recursive applications (Maximum path sum in a binary tree)
3. More complex recursive applications (Climb Stairs)

From the list above, we could clearly see that the questions cover almost all the related concept of recursion so that the user could examine their learning thoroughly.

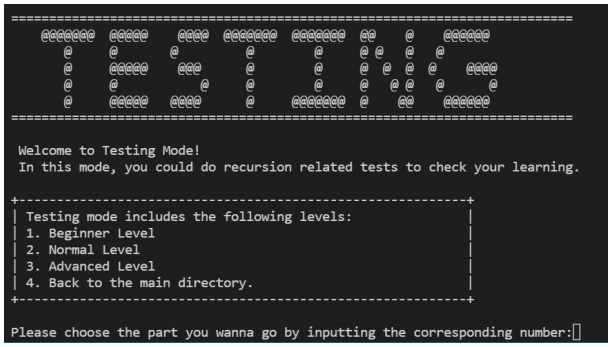
2.2 Interactive design

2.2.1 User interface design

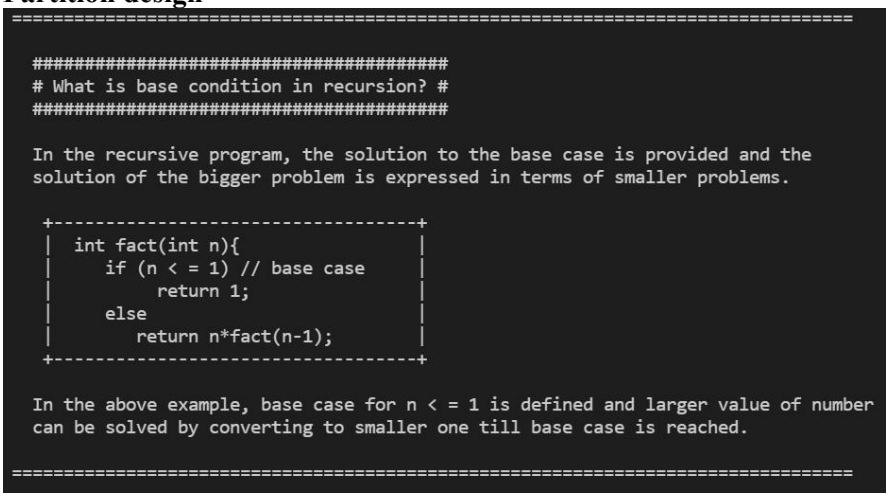
2.2.1.1 Main screen



When the user first enter the system, we will show such a welcom screen for the user. This made the system more intersting and more friendly to the user. Besides, both the learning mode and testing mode have their own welcome screen for the users (Show as below).



2.2.1.2 Partition design



Since in our system there is only 2 colors(black and white), the users may feel confused when they need to devide different parts. To improve the user experience, we apply partition design in our system. That is, we put different content in diffrent boxes and focus on page layout(as you can see in the above picture) so that it is clear for user to understand the content and they do not need to spend more time on concluding the key points on their own.

2.2.2 Input validation

```
+-----------------------------------------------------------+
| Testing mode includes the following levels:               |
| 1. Beginner Level                                         |
| 2. Normal Level                                           |
| 3. Advanced Level                                         |
| 4. Back to the main directory.                            |
+-----------------------------------------------------------+

Please choose the part you wanna go by inputting the corresponding number:5
Wrong input!
Please input the valid number agian: 0
Wrong input!
Please input the valid number agian: 1

=========================================================================

Welcome to beginner testing mode!
This is the mode for the beginners to check the learning for the very basic concepts.
```

Everytime the user input something, the system will check if it is valid. And if the input is invalid, the system will show an "Invalid input" notification and ask the user to input a valid message. The system will go to the next step only when the input is valid.

2.2.3 Flexible operation options

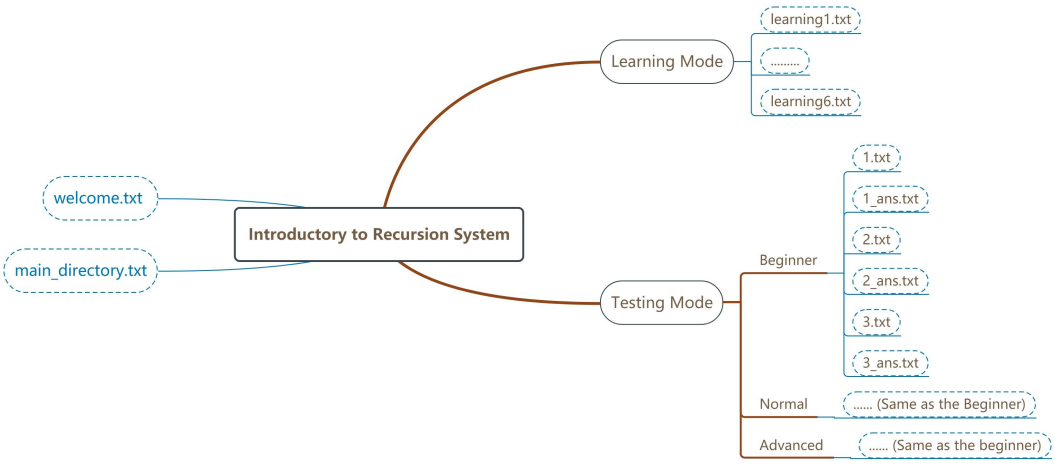
We provide flexible operation option for users to choose. For example, in learning mode, after the user learn one concept, the system will provide 2 choice to the user: directly go to the next part or go back to the learning mode main screen(Show as below)

```
+-----------------------------------+
| Please select the next step:      |
| 1. Go back to learning mode table |
| 2. Directly go to the next part   |
+-----------------------------------+
```

Besides, we also provide with similar flexible options to users in other parts so that the user can feel free to switch between different interface and mode.

2.3 File organizing

In our project, all the question and concept content were store in txt file. To make it easier to manage the files, we store them in different folders for different purposes(Show as below).



3. Code

3.1 Scalability

3.1.1 Modularization

To improve Scalability of our code, we define different functions.
For learning mode, we define the following functions:

```
//This is the function to show the corresponding concepts according to the user's choice.
void open_learning_file(int choice){···

//This is the function to showing the main table of learning mode
void leaning_mode_table(){···
```

For testing mode, we define the following functions:

```
//This is the function to get the file name of the answer file according to the question file.
string get_answer_file_name(string fileName){···

//This is the function to show the question and check user's answer
void do_the_test(string fileName, int choice_dir, int choice_file){···

//This is the function to go to the corresponding question file according to the user's input.
void open_testing_file(int choice_dir, int choice_file){···

//This is the function to go to the corresponding level directory according to the user's input
void open_testing_directory(int choice){···


//This is the mode to do some recursion related questions.
void testing_mode_table(){···
```

These functions realize the modularization of our code. It minimizes the impact of future expansion functions on the whole project.

3.2 Neatness

3.2.1 Common functions

Since our project needs read the contents of the TXT file for several times, we wrote shareable functionality "*get_txt_content*" and "*print_txt_content*" so that it could directly call theses two functions when the reading is needed. This avoids redundant code.

```
void main_directory();


/*
    Belows are common functions used in both Testing Mode and Learning mode
*/

void print_txt_content(string fileName){···

string get_txt_content(string fileName){···
```