

# Introduction to Computers and Programming Final-exam 20 2016/1/13 Time: 2 hrs

- ※Please create a new folder. Name the folder as: Student ID-Name (XXXXXXX-○○○). Inside the folder, your file format will be Q\_1.c, Q\_2.c, etc. There will ONLY be a total of 4 .c files in your folder (wrong file name or format will cause score deductions).
- ※No Internet. No discussions.
- ※The class is for C language, so do not use C++.
- ※If any of your program cannot be compiled, you will get zero score for the question.
- ※Before you use any variables, make sure you have assigned values to them. Some IDE's, such as Dev-C++, may not automatically initialize the variables.
- ※Your programs will be checked (by a tool) for the programming integrity. Be honest with your own works.
- ※Your output must comply with the sample output format.
- ※Please sign in the computer with your own account of CS Computer Center if you have it, and save your codes in your own space in case of something bad happens on your computer.

## 1. Save the world

Though you overcame many challenges in the past, yet these zombies keep coming back. Please grab your sword again, fight, and save the world!!

Please use skill\_Game\_framework.c and implement functions below:

- I. int Level1\_Trans\_nearest\_integer(float x)
- II. int Level2\_Get\_even\_num(int a,int b)
- III. int Level3\_check\_IsWinThisGame(int input,int answer,int digits,int \*A\_count,int \*B\_count)
- IV. void Level4\_permute(char \*text, int now\_pos, int final\_pos, char \*\*permutation\_list, int \*now\_count)

Note. You can only write Level1~Level4 functions, and add some new function. You are **NOT** allowed to modify int main() function or Level1~Level4 functions' arguments.

(Your points only depend on Level1~Level4 function codes)

```

***---Welcome to the Game World!---***
***-----level 1-----***
Write a program that asks the user to type a float number and rounds it to the nearest integer.
type a float number: 2.745
Your Ans: 3
***-----***

***-----level 2-----***
Write a program that ask the user to type 2 numbers( $0 \leq a \leq b \leq 2147483647$ ) and then calculate how many even numbers between a and b (include a and b).
type 2 numbers: 18 31
Your Ans: 7
***-----***

***-----level 3-----***
We want you to design a "Number Riddle" game. In this game, the player types 4 digits number (0~9, no repetition) and then he/she will get an output (OAXB).

The "A" means the value and position are both correct, and the "B" means only value is correct. The game will proceed until the player win the game (when he get the 4A0B).
What's your input number: 2341
You get the 0A4B!!
What's your input number: 3214
You get the 2A2B!!
What's your input number: 1234
You win!!
***-----***

***-----level 4-----***
Please write a program to get all permutations of a given char array, the length of input is between 1 and 7.
type text: abc
abc
acb
bac
bca
cba
cab
Total: 6
***-----***

***-----***
*~Congratulation!! Mission Completed!!~*
***-----***

Process returned 0 (0x0)   execution time : 45.261 s
Press any key to continue.

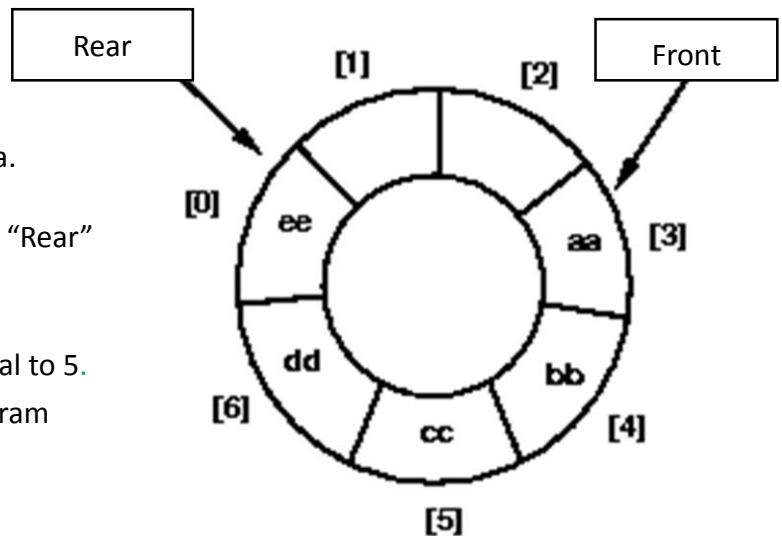
```

## 2. Circular Queue

A Circular Queue is another way to make use of queues. It allows you to reuse the place available after deleting old data.

The “Front” is the index of start and “Rear” is the tail of the data set.

Implement a circular of the size equal to 5.  
There are some functions your program should have:



**Hint:**

- Messages should be readable. ( do something to stop the program temporarily )**
- Every requirement will be tested independently, so implement as many functions as possible.**
- When the queue is empty, the ‘front ‘ is ‘ 0 ‘ and ‘Rear’ should be ‘-1’**

### 1. Enqueue

- Insert a character into the queue.
- Show message “ The queue is full.” when the queue is already full.

### 2. Dequeue

- Delete the data in the place that “**Front**” is pointing at.
- Print” Queue will underflow.” when the queue is empty and prevent it from underflow.

### 3. Display

- Show “Front” value, “Rear” value, and the current queue size as shown in the example below.
- If the queue is empty, show message” Queue is empty”
- Show the current data in the queue as shown in the example below.

### 4. Exit

- Exit the program.

## Enqueue

```

1. Enqueue.
2. Dequeue.
3. Display.
4. Exit.

```

What are you going to do? - 1  
insert:a

```

1. Enqueue.
2. Dequeue.
3. Display.
4. Exit.

```

What are you going to do? - 1  
insert:b

What are you going to do? - 3  
front: 0  
rear: 1  
Queue Size: 5  
Data Size:2

0	1	2	3	4
a	b			

↓ Until queue is full

What are you going to do? - 3  
front: 0  
rear: 4  
Queue Size: 5  
Data Size:5

0	1	2	3	4
a	b	c	d	e

What are you going to do? - 1  
Queue is full.

## Dequeue

```

1. Enqueue.
2. Dequeue.
3. Display.
4. Exit.

```

What are you going to do? - 2

What are you going to do? - 3  
front: 1  
rear: 4  
Queue Size: 5  
Data Size:4

0	1	2	3	4
	b	c	d	e

## Delete to empty

What are you going to do? - 2



Until queue is empty

What are you going to do? - 3  
front: 0  
rear: -1  
Queue Size: 5  
Data Size:0

Queue is empty

What are you going to do? - 2  
Queue will underflow

## Example1

## Exame2 : extend queue size and display format

What are you going to do? - 3  
front: 2  
rear: 0  
Queue Size: 5  
Data Size:4

0	1	2	3	4
d		a	b	c



### 3. Calculate Scores

Write a C program that searches through a list file and several score files for **the best student** and **the best class**.

First, the user is asked to input the path to a file that contains the class names.

For each **class name**, the corresponding **score file** name is **className.txt**

The first line of each **score file** has two numbers:

**The first** is the number of students in this class.

**The second** is the number of scores each student has.

The remaining lines are composed of **student id** and student scores.

**The best student** is defined as the one who has **the highest average score over all classes**, and **the best class** is defined as **the class for which all students performed the best on average**.

Now, you can use the information above to complete the C program.

#### Note:

1. Please round the average score **to the 2<sup>nd</sup> digit** after the decimal point
2. We will put your C program, the class list file, and the score files **in same directory**.
3. **student id** and **class name** length **cannot exceed 100 char**
4. **You have to follow the output format bellow!! (or you will get 0 point)**

Example:

```
Please input class list path:classList.txt
Best student: R0456032
Best student average score: 96.67
Best class: classB
Best class average score: 84.75
```

#### 4. Paragraph Decompression

You've already learned a method to compress a paragraph in *lab12*. **If you forget the implementation details, please read the reference file, called “*Compression.pdf*”.**

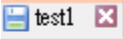
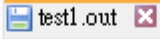

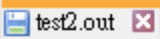
Now, **your job is to decompress a paragraph that is compressed by this method.**

In your program, first ask the user to input the filename of the compressed paragraph, and then decompress the file to another file. If the input filename is abcdefg, the output filename must be abcdefg.out.

##### Program window

```
Please enter the filename of the compressed paragraph: test1
The decompression paragraph is generated in file test1.out!
```

##### Decompression results (A compression paragraph to a decompression paragraph)

 test1	 test1.out
<pre>1 Amy my 2 amy My MY AMY 6 1'6</pre>	<pre>1 Amy my Amy amy My MY AMY my my'Amy</pre>
 test2	 test2.out
<pre>1 I love you. 2 You 3 him, 3 and he loves 4! 4 He 3 her. &gt;&lt; 5 However...she 4 11 9 12.</pre>	<pre>1 I love you. 2 You love him, 3 and he loves him! 4 He loves her. &gt;&lt; 5 However...she loves you and I.</pre>