PRINCE OF SONGKLA UNIVERSITY

FACULTY OF ENGINEERING

Final Examination: ภาคการศึกษาที่ 2/2553 Lecturer: ดร. อารีย์, อ.สุธน และ อ.จักรพันธ์

Date: 24 กุมภาพันธ์ พ.ศ. 2554 Time: 09.00 – 12.00 น.

Subject: 241-207 Data Structures and Computer Programming Techniques Room: S101, S104

ทุจริตในการสอบ มีโทษขั้นต่ำ คือ ปรับตกในรายวิชาที่ทุจริต และพักการเรียน 1 ภาคการศึกษา

อ่านรายละเอียดของข้อสอบ และคำแนะนำให้เข้าใจก่อนเริ่มทำข้อสอบ

อนุญาต: เครื่องเขียน

ไม่อนุญาต: หนังสือ และเครื่องคิดเลข

รหัสนักศึกษา .....

ชื่อ-สกุล .....

เวลา 3 ชั่วโมง (180 นาที)

คำแนะนำ:

- ข้อสอบมี 11 หน้า (ไม่รวมหน้าปก) มีทั้งหมด 8 ข้อ คะแนนรวม 180 คะแนน (คิดเป็น 40%)
- คำตอบส่วนใดอ่านไม่ออก จะถือว่าคำตอบนั้นผิด
- เขียนชื่อ (ไม่ต้องเขียนนามสกุล) และรหัสนักศึกษา ในทุกหน้า
- พยายามทำทุกข้อ และคำตอบทั้งหมดให้ทำในข้อสอบชุดนี้ หากตอบไม่พอให้ต่อด้านหลังข้อสอบ
- อ่านคำสั่งในแต่ละข้อให้ชัดเจนว่า เขียนโปรแกรมบางส่วน เขียนฟังก์ชัน หรือเขียนทั้งโปรแกรม รวมไปถึงข้อกำหนดเพิ่มเติม และหมายเหตุในข้อนั้นๆ
- การเขียนโปรแกรมในแต่ละข้อ อาจจะไม่ต้องเขียนตามคำสั่งย่อยทั้งหมด แต่คะแนนจะลดลงตามส่วน
- การเขียน code จะต้องตั้งชื่อตัวแปรให้เหมาะสม และมี comment ในจุดสำคัญต่างๆ โดยให้ทั้งหมดเป็นไปตามหลักการเขียนโปรแกรมที่ดี

สำหรับอาจารย์

ข้อ	1 (20)	2 (10)	3 (20)	4 (40)	5 (20)	6 (10)	7 (30)	8 (30)	รวม (180)
คะแนน									

Abstract Data Type (20 คะแนน)

1. จง**อธิบาย**ความหมาย ของคำต่อไปนี้ พร้อม**ยกตัวอย่าง**ประกอบ

   1.1. Abstract Data Type : .....

   .....

   .....

   1.2. Data : .....

   .....

   1.3. Operations : .....

   .....

   1.4. Information hiding : .....

   .....

   1.5. Implementation : .....

   .....

File (10 คะแนน)

2. จากตัวแปรที่กำหนดให้ จงเขียนโค้ดตามคำสั่งต่อไปนี้

   ```
   FILE* finput_ptr;
   ```

   2.1. กำหนดตัวแปร `finput_ptr` สำหรับเปิดไฟล์ input.txt แบบอ่านและเขียน

   .....

   2.2. ทดสอบตัวแปร `finput_ptr` ว่าเปิดแฟ้มข้อมูลสำเร็จ โดยถ้าเปิดแฟ้มไม่สำเร็จให้พิมพ์คำว่า Cannot open file. และออกจากโปรแกรม

   .....

   .....

   .....

2.3. อ่านข้อมูลมาหนึ่งแถวของแฟ้มข้อมูล (text file) ที่ละตัวอักษร ให้กับตัวแปรชื่อ line โดยวนลูปอ่านจนกว่าจะหมดแถว (เจอค่า `\n`) หรือ สิ้นสุดแฟ้มข้อมูล (`EOF`)

.....

.....

.....

.....

.....

.....

.....

.....

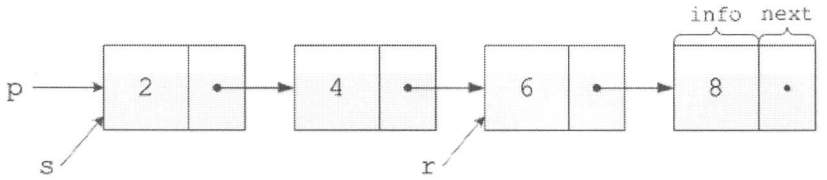
2.4. ปิดแฟ้มข้อมูล ของตัวแปร `finput_ptr`

.....

Linked List

(20 คะแนน)

3. จากรูปของ Linked List ที่กำหนดให้ต่อไปนี้



จงเขียนรูปของ Linked List ที่เปลี่ยนไป เมื่อโปรแกรมทำงานจนสิ้นสุดบรรทัดสุดท้ายของโค้ดข้างล่าง (รูปเดียว)

```
r->next->next = p;
s = p->next;
(r->info)++;
p = r->next;
r->next = NULL;
```

คำตอบข้อ 3 ...

File & Linked List **(40 คะแนน)**

4. จากโค้ดที่กำหนดให้ จงเขียนฟังก์ชันที่ขาดหายไป โดยมีข้อกำหนดเพิ่มเติมว่า ฟังก์ชัน `read_song_db` ต้องเรียกใช้ฟังก์ชัน `create_listnode` และ `gfopen` อย่างเหมาะสม

```
typedef struct {
    int song_id;
    char file_name[25];
    int p_count; // Count number of playing
    int percent_p; // Percent played per total
} SONG;

typedef struct listnode {
    SONG *song;
    struct listnode *next_ptr;
} LISTNODE;

typedef LISTNODE *LISTP;

LISTP create_listnode( SONG *song ) {
    LISTP new_node = ( LISTP )malloc( sizeof( LISTNODE ) );
    new_node->song = song; // Linked to SONG node
    return new_node;
}

FILE *gfopen(char *file_name, char *mode) // Graceful open file
{
  FILE *fp;
  if ( ( fp = fopen( file_name, mode) ) == NULL ) {
     printf( "Cannot open file." );
     exit(1);
  }
  return fp;
}

int main(){

    // Read list of song from file
    LISTP song_lst = read_song_db();

    // Calculate percent played and update to file
    update_song_db ( song_lst );
}
```

โค้ดข้างต้นจะอ่านข้อมูลการเล่นเพลงในเครื่องเล่น mp3 ซึ่งบันทึกอยู่ในรูปไฟล์ไบนารีชื่อ song_db.dat ขึ้นมาทั้งหมด เพื่อคำนวณหาความบ่อยในการเล่นแต่ละเพลง แล้วบันทึกข้อมูลกลับลงในไฟล์เดิม โดยมีรายละเอียดดังนี้

- โปรแกรมจะอ่านข้อมูลในไฟล์ทั้งหมดขึ้นมาเก็บในโครงสร้างข้อมูลลิงค์ลิส์ เพื่อหาจำนวนครั้งที่เล่นรวมทุกเพลง
- ความบ่อยในการเล่น คิดจากร้อยละของจำนวนครั้งที่เล่นเพลงนั้น ต่อจำนวนครั้งที่เล่นรวมทุกเพลง เช่น เพลง my_love.mp3 เล่นไปทั้งหมด 5 ครั้ง จากการเล่นทุกเพลงในเครื่องรวมกัน 20 ครั้ง ดังนั้น ความบ่อยจะเท่ากับ (5 / 20) x 100 = 25 เปอร์เซ็นต์
- การอ่านเขียนไฟล์ใช้โครงสร้างข้อมูล SONG โดยมี `song_id` กำหนดลำดับการบันทึกข้อมูลลงในไฟล์
- ภายในไฟล์ song_db.dat สามารถมีข้อมูล SONG ได้หลายชุด

ตัวอย่างข้อมูลภายในไฟล์ song_db.dat ก่อนและหลังรันโปรแกรม

	Before	After
song_id	1	1
file_name	my_love.mp3	my_love.mp3
play_count	5	5
percent_play	0.0	25.0
	2	2
	my_hump.mp3	my_hump.mp3
	12	12
	0.0	60.0
	3	3
	tonight.mp3	tonight.mp3
	3	3
	0.0	15.0

ให้ตอบในหน้าถัดไป

Stack & Queue (30 คะแนน)

5. จงเขียนฟังก์ชันชื่อ `match_parenthesis` เพื่ออ่านอักขระจากแป้นพิมพ์ขนาดความยาวไม่เกิน 80 ตัวอักษร จากนั้นให้ตรวจสอบดูว่าอักขระที่อ่านมานั้นประกอบด้วยวงเล็บเปิดและปิดที่ครบคู่กันหรือไม่ กรณีที่วงเล็บเปิด-ปิด สัมพันธ์กัน (ครบคู่) ฟังก์ชันนี้จะส่งคืนค่า TRUE มิฉะนั้น จะคืนค่า FALSE (20 คะแนน)

```
//Definition of stack data structure
typedef struct node {
    char data;
    struct node* link;  // pointer to next node
} STACK_NODE;

// Stack head
typedef struct {
    int count;
    STACK_NODE* top; //pointer to stack top node
} STACK;
typedef STACK* STACKP; //pointer to stack

// Create a new stack
STACKP createStack();

// Push stack
int pushStack( STACKP stk, char n );

// Pop stack
int popStack( STACKP stk, char* n );

// Empty stack checking
int isEmpty( STACKP stk );
```

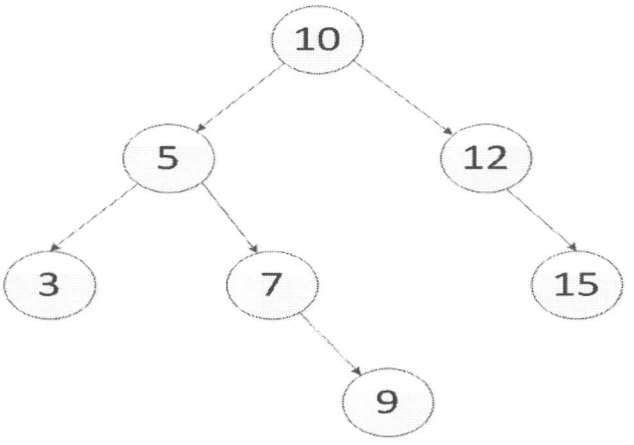
หมายเหตุ น.ศ. สามารถเรียกใช้โครงสร้างข้อมูล stack จากต้นแบบฟังก์ชันที่กำหนดให้ได้เลย

ให้เขียนคำตอบในหน้าถัดไป

6. จงวาดเฉพาะไดอะแกรม (ไม่ต้องอธิบาย) เพื่อแสดงโครงสร้างและส่วนประกอบที่สำคัญของคิว (Queue) ที่ implement โดยใช้ Singly Linked List (มี pointer ตัวเดียว) พร้อมทั้งแสดงทิศทางของข้อมูลที่เข้าและออกจากคิว (10 คะแนน)

Tree (30 คะแนน)

7. จากโครงสร้างข้อมูลที่กำหนดให้ด้านล่าง จงตอบคำถามต่อไปนี้



7.1. ทฤษฎีเกี่ยวกับ Tree (6 คะแนน)

ความสูงของ tree คือ .....

Children ของโหนด 5 คือ .....

Leaf nodes ทั้งหมด ได้แก่ .....

7.2. จงหาผลลัพท์ของลำดับการเข้าถึง Tree แบบต่อไปนี้ (14 คะแนน)

Inorder : .....

Postorder : .....

7.3. จากส่วนของโค้ดที่กำหนดให้ จงเขียนฟังก์ชันเรียกตัวเอง (Recursive function) ชื่อ `height` สำหรับหาความสูงของ Tree โดยรับพารามิเตอร์เป็น `TREE` และส่งค่ากลับเป็นความสูงของ Tree (10 คะแนน)

```
struct treenode {
    struct treenode *leftptr;
    int data;
    struct treenode *rightptr;
};

typedef struct treenode TREENODE;
typedef TREENODE* TREE;
```

```
int height ( TREE t ) {









}
```

Big-O, Search, Sort **(30 คะแนน)**

8. จงตอบคำถามต่อไปนี้

8.1. จากฟังก์ชันในการวัดประสิทธิภาพของอัลกอริทึมที่กำหนดให้ จงเรียงลำดับของค่า Big-O ต่อไปนี้ โดยเรียงลำดับจากประสิทธิภาพด้อยที่สุดไปประสิทธิภาพดีที่สุด โดยเรียงลำดับจากฟังก์ชันต่อไปนี้

$O(n^3)$, $O(n)$, $O(n \log n)$, $O(n^2)$, $O(1)$, $O(\log n)$ (10 คะแนน)

คำตอบ $O(n^3)$ > _____ > _____ > _____ > _____ > _____

8.2. ทฤษฎีการ Search (10 คะแนน)

8.2.1. Worst case ในการทำ Linear Search เกิดขึ้นเมื่อใด

.....

8.2.2. ประสิทธิภาพ (Big-O) ของ Linear Search เมื่อข้อมูลถูกจัดเรียงแล้ว คืออะไร

.....

8.2.3. ประสิทธิภาพของ Binary Search แบบทำซ้ำและแบบ Recursive แตกต่างกันหรือไม่ อย่างไร

.....

8.2.4. ข้อมูลที่ใช้ในการค้นหาแบบ Binary Search จะต้องจัดเรียงข้อมูลก่อนหรือไม่ อย่างไร

.....

8.2.5. เปรียบเทียบสองอัลกอริทึมในการเรียงข้อมูล Linear และ Binary Search อะไรเร็วกว่า ทำไมถึงเป็นเช่นนั้น

.....

8.3. จงเรียงลำดับข้อมูลที่กำหนดให้ต่อไปนี้ โดยใช้อัลกอริทึม Selection Sort (10 คะแนน)

เมื่อกำหนดค่าเริ่มต้นในอาร์เรย์เป็น 12 18 27 25 4 2

จงหาค่าสุดท้ายในอาร์เรย์หลังการเรียงข้อมูลในแต่ละรอบ

รอบ 1 .....

รอบ 2 .....

รอบ 3.....

รอบ 4.....

รอบ 5.....

/*---------- End of Final Exam: Good Luck ☺ ----------*/