ชื่อ _____ รหัส _____

มหาวิทยาลัยสุโขทัยธรรมาธิราช
คณะวิศวกรรมศาสตร์

สอบปลายภาค: ภาคการศึกษาที่ 2 **ปีการศึกษา**: 2554

วันที่สอบ: 23 กุมภาพันธ์ 2555 **เวลาสอบ**: 09.00 – 12.00 น.

ห้องสอบ: R201

ผู้สอน: ดร.อารีย์ ธีรภาพเสรี ภาควิชาวิศวกรรมคอมพิวเตอร์

รหัสและชื่อวิชา: **241-207** Data Structure and Computer Programming Techniques

ทุจริตในการสอบมีโทษขั้นต่ำคือ ปรับตกในรายวิชาที่ทุจริตและพักการเรียน 1 ภาคการศึกษา

คำสั่ง: อ่านรายละเอียดของข้อสอบ และคำแนะนำให้เข้าใจก่อนเริ่มทำข้อสอบ

อนุญาต: เครื่องเขียนต่างๆ เช่น ปากกา หรือดินสอ เข้าห้องสอบ

ไม่อนุญาต: หนังสือ หรือเครื่องคิดเลขเข้าห้องสอบ และ**เอกสารใดๆ เข้าและออกห้องสอบ**

เวลา: 3 ชั่วโมง (180 นาที)

คำแนะนำ

- ข้อสอบมี 7 หน้า (*รวมหน้าปก*) แบ่งออกเป็น 4 ตอน คะแนนรวม 40 คะแนน (คิดเป็น 35%)
- **<u>เขียนคำตอบในข้อสอบ คำตอบส่วนใดอ่านไม่ออก จะถือว่าคำตอบนั้นผิด</u>**
- อ่านคำสั่งในแต่ละข้อให้เข้าใจก่อนลงมือทำ
- หากข้อใดเขียนคำตอบไม่พอ ให้เขียนเพิ่มที่ด้านหลังของหน้านั้นเท่านั้น

ตอน	1 (10)	2 (10)	3 (10)	4 (10)	รวม (40)
คะแนน					

นักศึกษารับทราบ ลงชื่อ .....

ชื่อ _____ รหัส _____

Stack & Queue (10 คะแนน)

1. จากโครงสร้างข้อมูลและฟังก์ชันที่กำหนดให้ต่อไปนี้ ให้น.ศ.เขียน**ฟังก์ชัน** evaluate() ที่ใช้หลักการของ Stack มาช่วยในการคำนวณหาผลลัพธ์ของ Postfix Expression (ของตัวเลข 0-9) เช่น postfix expression `6 2 * 3 -` จะได้ผลการคำนวณเป็น 9 (คำนวณจาก (6 * 2) – 3)

   ฟังก์ชัน evaluate() return 0 ถ้า expression ไม่สมบูรณ์ ไม่สามารถคำนวณผลลัพธ์ได้ และ return 1 ถ้าผลการคำนวณถูกต้อง โดยผลการคำนวณที่ถูกต้องคือ เมื่ออ่านค่า expression จนครบทุกตัวแล้ว มีค่าตัวเลขอยู่ใน stack เพียง 1 ค่า ซึ่งเป็นค่าผลลัพธ์ที่ได้จากการคำนวณนั่นเอง

   หมายเหตุ น.ศ.สามารถเรียกใช้งานฟังก์ชัน push(), pop(), isOperand() และ isOperator() ได้โดยไม่จำเป็นต้องเขียนนิยามรายละเอียดของฟังก์ชัน โดยฟังก์ชันมีผลการทำงานดังที่ได้เขียนกำกับไว้

```
typedef struct node
{  int data;
   struct node* link;
} STACK_NODE;

typedef struct
{ int count;
  STACK_NODE* top;
} STACK;
// ฟังก์ชัน push และ pop จะคืนค่า 1 ถ้า push และ pop สำเร็จ ถ้าไม่สำเร็จจะคืนค่า 0
int push(STACK *stk, int n); //ฟังก์ชันเก็บค่า n ลงใน stack stk
int pop(STACK *stk, int* n); //ฟังก์ชันอ่านค่าจาก stack stk มาเก็บในตัวแปร n
int isOperand(char c); // คืนค่า 1 ถ้า c เป็นตัวเลข และคืนค่า 0 ถ้าไม่ใช่
int isOperator(char c); // คืนค่า 1 ถ้า c เป็นเครื่องหมายคณิตศาสตร์ และคืนค่า 0 ถ้าไม่ใช่
```

```
// stk เป็น pointer ที่ชี้ไปยัง stack ว่าง และ expr เก็บค่า expression ที่ต้องการคำนวณ
int evaluate(STACK *stk, char expr[100])
{
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
```

ชื่อ _____ รหัส _____

}

TREE (10 คะแนน)

2. จงใช้โครงสร้างข้อมูล Binary Tree ต่อไปนี้ เพื่อเขียนฟังก์ชันในข้อ 2.1 และข้อ 2.2

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

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

ชื่อ _____ รหัส _____

2.1 จงเขียนฟังก์ชัน int countLeafNode(TREE t) ที่ส่งค่ากลับเป็นจำนวนของ Leaf Node ของ t (5 คะแนน)

// ใช้ static int count เพื่อให้สามารถนับจำนวนต่อเนื่องจากการเรียก ครั้งก่อนได้ เมื่อเรียกฟังก์ชันเดิมซ้ำอีก

```
int countLeafNode(TREE t) {
      static int count = 0;
```

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

}

2.2 จงเขียนฟังก์ชัน int minNode(TREE t) เพื่อหาค่า data ที่มีค่าน้อยที่สุดใน TREE t (5 คะแนน)

```
int minNode(TREE t) {
```

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

}

ชื่อ _____ รหัส _____

Search (10 คะแนน)

3. จากฟังก์ชัน fooSearch ที่กำหนดให้ จงตอบคำถามต่อไปนี้

```
int fooSearch(int array[], int left, int right, int key)
{
   int mid;
    if ( _______(1)_____________)    /* ไม่มีข้อมูลให้ค้นหา */

      return -1;
    mid = (left+right)/2;
    if ( ______(2)_____________)
      return mid;
    if (array[mid]) >key)
      return( fooSearch(array,left, mid-1, key) );
  else
      return( fooSearch( _________(3)_____________) );
}
```

3.1 จงเติมส่วนที่ขาดหายไป เพื่อให้ฟังก์ชัน fooSearch สามารถตรวจสอบได้ว่า ค่าในตัวแปร key มีอยู่ในตัวแปร array[] หรือไม่ (3 คะแนน)

ตอบ 1) .....

2) .....

3) .....

3.2 ฟังก์ชัน fooSearch ใช้ algorithm ในการค้นหาข้อมูลแบบใด (2 คะแนน)

ตอบ .....

.....

3.3 ฟังก์ชัน fooSearch มีค่า O() ในแต่ละกรณี เป็นอย่างไร (3 คะแนน)

ตอบ Best case: .....

Worst case: .....

4. จงหา Big-O ของและอัลกอริธึมดังต่อไปนี้ (2 คะแนน)

a) อัลกอริธึมในการค้นหาแบบที่ 1 **ใช้เวลา** 5n + nlogn

.....

b) อัลกอริธึมในการค้นหาแบบที่ 2 **ใช้เวลา** n^2+20n+500

.....

ชื่อ _____ รหัส _____

Sort (10 คะแนน)

5. จงเขียนผลลัพธ์ที่เปลี่ยนไปในแต่ละรอบของการเรียงลำดับข้อมูลในแต่ละ[redacted]อัลกอริธึมดังนี้คือ Insertion sort และ Selection sort โดยข้อมูลที่ต้องทำการเรียงคือ 9 4 2 7 5 3

```
ตัวอย่างการแสดงผลลัพธ์ในแต่ละรอบ
ข้อมูล: 3 5 2 7 4 2
รอบที่ 1 : 3 2 5 4 2 7
รอบที่ 2 : 2 3 4 2 5 7
รอบที่ 3 : 2 3 2 4 5 7
รอบที่4 : 2 2 3 4 5 7
รอบที่ 5 : 2 2 3 4 5 7
หยุด
```

5.1 Insertion sort (3 คะแนน)

ข้อมูลที่ต้องทำการเรียงคือ: 9 4 2 7 5 3

.....

.....

.....

.....

.....

.....

.....

.....

5.2 Selection sort (3 คะแนน)

ข้อมูลที่ต้องทำการเรียงคือ: 9 4 2 7 5 3

.....

.....

.....

.....

.....

.....

.....

.....

6. จงเปรียบเทียบประสิทธิภาพ ในแง่ของเวลา (Best case, Worse case, Average case) ที่ใช้ในการทำงานในรูปแบบ O() ของ sorting algorithm ดังต่อไปนี้ (4 คะแนน)

- Bubble sort
- Quick sort

.....

.....

.....

.....

.....

.....

.....

.....