PRINCE OF SONGKLA UNIVERSITY

FACULTY OF ENGINEERING

Midterm Examination: ภาคการศึกษาที่ 1/2554 **Date:** 2 สิงหาคม 2554 09.00-11.00 น.

Subject Number: 241-207 **Room:** R200, S203, A403

Subject Title: Data Structures and Computer Programming Techniques

ทุจริตในการสอบ มีโทษขั้นต่ำ คือ ปรับตกในรายวิชาที่ทุจริต และพักการเรียน 1 ภาคการศึกษา

อ่านรายละเอียดของข้อสอบ และคำแนะนำให้เข้าใจก่อนเริ่มทำข้อสอบ

รหัสนักศึกษา .....

ชื่อ-สกุล .....

สำหรับอาจารย์

ข้อ	คะแนน
1	
2	
3	
4	

รายละเอียดของข้อสอบ:

เวลา 2 ชั่วโมง (120 คะแนน: 120 นาที)

เอกสารมีทั้งหมด 7 หน้า (ไม่รวมหน้านี้)

คำถามจำนวน 4 ข้อ

สิ่งที่สามารถนำเข้าห้องสอบได้:

อนุญาต: เครื่องเขียน

ไม่อนุญาต: หนังสือ และเครื่องคิดเลข

คำแนะนำ:

- เขียนชื่อ (ไม่ต้องเขียนนามสกุล) และรหัสนักศึกษา ในทุกหน้า
- พยายามทำทุกข้อ และคำตอบทั้งหมดให้ทำในข้อสอบชุดนี้
- คำตอบส่วนใดอ่านไม่ออก จะถือว่าคำตอบนั้นผิด
- อ่านคำสั่งในแต่ละข้อให้ชัดเจนว่า เขียนโปรแกรมบางส่วน เขียนฟังก์ชัน หรือเขียนทั้งโปรแกรม รวมไปถึงข้อกำหนดเพิ่มเติม และหมายเหตุในข้อนั้นๆ
- การเขียนโปรแกรมในแต่ละข้อ อาจจะไม่ต้องเขียนตามคำสั่งย่อยทั้งหมด แต่คะแนนจะลดลงตามส่วน
- การเขียน code จะต้องตั้งชื่อตัวแปรให้เหมาะสม และมี comment ในจุดสำคัญต่างๆ โดยให้ทั้งหมดเป็นไปตามหลักการเขียนโปรแกรมที่ดี

อ.สุธน: ผู้ออกข้อสอบ

ข้อที่ 1 ความรู้พื้นฐาน (20 คะแนน)

1.1 จากอาเรย์ a จงเขียนส่วนของโค้ด เพื่อแสดง เลขจำนวนเต็มที่ติดลบ (5 คะแนน)

```
int a[5] = {3, 2, 7, -4, 6};
```

.....

.....

.....

.....

1.2 จากข้อมูลที่กำหนดให้ จงเขียนคำสั่งตามที่กำหนดในแต่ละข้อย่อย (5 คะแนน)

```
int x = 0;
int *ptr;
```

ให้ ptr อ้างอิงไปยังตัวแปร x

.....

เพิ่มค่าตัวแปร x ขึ้นอีก 1 ผ่านตัวแปร ptr

.....

1.3 จงเขียนแผนภาพแสดงค่าของตัวแปร และการเชื่อมโยงของตัวแปรทุกตัวในหน่วยความจำ เมื่อโปรแกรมทำงานจนสิ้นสุดบรรทัดสุดท้ายของโค้ดข้างล่าง (10 คะแนน)

```
int a = 7, *p, b[5] = {2, 1}, *s[2];
s[0] = &a;
s[1] = b;
*(s[1] + 2) = 7;
p = b + 4;
(*p)++;
```

ข้อที่ 2 ข้อความและตัวอักษร (30 คะแนน)

จงเขียนฟังก์ชันเพื่อตัดส่วนของข้อความจาก ข้อความ s โดยใช้ตัวอักษร a และ ตัวอักษร b เป็นขอบเขตในการตัดข้อความ โดยมีข้อกำหนดพื้นฐานดังนี้

- ต้นแบบของฟังก์ชัน

```
char* scut(char *s, char a, char b);
```

- ฟังกัน scut จะต้องมีการเรียกใช้ฟังก์ชัน cfind อย่างเหมาะสม เพื่อหาพอยเตอร์ที่ชี้ไปยังตัวอักษรที่ต้องการค้นหา

```
char* cfind(char *s, char key);
```

- ฟังก์ชันคืนค่าเป็นส่วนข้อความ (จากข้อความ s) ที่ทำการตัดเรียบร้อยแล้ว
- ห้ามเรียกใช้ฟังก์ชันมาตรฐานเกี่ยวกับข้อความ เช่น `strlen` หรือ `strcpy` (หากผิดเงื่อนไขนี้ หัก 10 คะแนน)
- ห้ามใช้เครื่องหมาย [] ในโค้ด (หากผิดเงื่อนไขนี้ หัก 10 คะแนน)
- ข้อความที่ถูกตัดจะถูกตัดตั้งแต่ตัวอักษร a ถึง b หรือ b ถึง a ก็ได้ หากไม่เจอ a หรือ b ในข้อความ ให้ผลลัพธ์การตัดคือข้อความเดิม
    - scut("Hello world", 'e', 'r') => "ello wor"
    - scut("Hello world", 'r', 'e') => "ello wor"
    - scut("Hello world", 'w', 'x') => "Hello world"
    - scut("Hello world", 'x', 'd') => "Hello world"

3.1 ส่วนนิยามฟังก์ชัน `cfind` (10 คะแนน)

```
char* cfind(char *s, char key){




}
```

2.2 ส่วนนิยามฟังก์ชัน scut (20 คะแนน)

```
char* scut(char *s, char a, char b){

    char *result = s;
    //ประกาศตัวแปรอื่นๆ ที่จำเป็นต้องใช้




    //หาพอยเตอร์อ้างอิงไปยังตัวอักษร แล้วปรับให้ a เป็นด้านหน้า และ b เป็นด้านหลัง
    //โดยใช้ประโยชน์จาก a > b หรือ b > a









    //ในกรณีที่หา a และ b เจอ ก็ให้ใช้ a เป็นจุดเริ่มต้นของข้อความ
    //เพิ่ม \0 หลัง b









    return result;
}
```


รหัสนักศึกษา .....

ข้อที่ 3 อาเรย์ พอยเตอร์และหน่วยความจำ (30 คะแนน)

โค้ดข้างล่าง เป็นโค้ดที่แสดงการประมวลผลกับอาเรย์ data จงใช้ข้อมูลในโค้ดตอบคำถามข้อ 3.1 และ 3.2

```
#include<stdio.h>
#include<stdlib.h>

#define LEN 5

int* foo(int data[], int len){
   int r[LEN];
   int *p = r;
   int *q = r + len;
   while(p < q){
      if(*p > 0)
          (*p)++;
      else
          *p = 0;
      p++;
   }
   return r;
}
int main(){
   int a[5] = {1, 4, 3, -2, 9};
   int i;
   int *r = foo(a, LEN);

   for(i = 0; i < LEN; i++){
      printf("%d ", r[i]);
   }
   printf("\n");

   return 0;
}
```

3.1 โค้ดข้างต้นมีความผิดพลาดในการจัดการหน่วยความจำหรือไม่ หากมีให้แก้ไข หรือเพิ่มเติม **ให้เขียนไว้ร่วมกับโค้ดด้านบน พร้อมคำอธิบายประกอบ (การให้คะแนนขึ้นอยู่กับคำอธิบายประกอบ)** (20 คะแนน)

หมายเหตุ หากมีการใช้งาน Dynamic Memory Allocation จะต้องมีการคืนค่าหน่วยความจำอย่างเหมาะสมด้วย

3.2 จงเขียนผลลัพธ์ที่ถูกแสดงเมื่อรันโปรแกรมข้างต้น (หลังจากแก้ไขข้อผิดพลาดแล้ว) (10 คะแนน)

.....

ข้อที่ 4 โครงสร้างข้อมูล (40 คะแนน)

จากโค้ดของที่กำหนดให้ จงเติมส่วนที่ขาดหายไป เพื่อรับ Polygon จำนวนหนึ่ง มาหาว่า Polygon รูปใดมีความยาวเส้นรอบวง (perimeter) มากที่สุด

โดยทั้งนี้ฟังก์ชันต่างๆ ต้องถูกเรียกใช้อย่างเหมาะสม

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>

typedef struct {
  int x;
  int y;
}Point;

typedef struct {
  int angle;
  Point *gons;
}Polygon;  //รูป n เหลี่ยม จะประกอบด้วยจุด n จุด โดยให้ถือว่า จุดสุดท้ายเชื่อมโยงมาจุดแรก

float distance(Point *p1, Point *p2);
void input_polygon(Polygon *p);
float perimeter(Polygon *p);
Polygon *find_max(Polygon ps[], int len);

main(){
  Polygon ps[5];
  Polygon *p;
  int i;

  for(i = 0; i < 5; i++){
     printf("P%02d: ", i + 1);
     input_polygon(ps + i);
  }

  p = find_max(ps, 5);
```

```
//แสดงหมายเลข index ของ Polygon ที่มีความยาวเส้นรอบวงสูงสุด (เริ่มจาก 0)
printf("Longest perimeter belongs to the polygon with index = %d\n",

                        );
```


```
}

float distance(Point *p1, Point *p2){          //หาระยะห่างระหว่างจุดสองจุด
   return sqrt(pow(p1->x - p2->x, 2) +
          pow(p1->y - p2->y, 2));
}

void input_polygon(Polygon *p){   //รับข้อมูลของ Polygon จากผู้ใช้
   int i;
   Point *tmp;

   scanf("%d", &(p->angle));
```


```
   for(i = 0; i < p->angle; i++){
      tmp = p->gons + i;
      scanf(" %d %d", &(tmp->x), &(tmp->y));
   }

}
```


รหัสนักศึกษา .....

```
float perimeter(Polygon *p){      //หาความยาวเส้นรอบวงของ Polygon ที่กำหนด
   float perim = 0;
   int i;
```


```
   return perim;
}

Polygon *find_max(Polygon ps[], int len){     //หารูปเหลี่ยมที่มีเส้นรอบวงสูงสุด
   Polygon *mp  = ps;
```


```
   return mp;
}
```


รหัสนักศึกษา .....