CP 111 Computer Programming (2/2554 Section B01-B02)

Handout 13

<u>หัวข้อ:</u>

- ทบทวน: สรุปหัวข้อที่ได้เรียนไปทั้งหมด
- ทบทวน: Linked lists
- ทบทวน: Recursion
- ทบทวน: เรื่องอื่นๆ

<u>ทบทวน: สรุปหัวข้อที่ได้เรียนไปทั้งหมด</u>

(1) **Basic programming in C** (Chapter 11- 14)

- Variables, Literals, Operators, Expression, Statement
- Conditional constructs: `if`, `if-else`, `switch` statements
- Iterative constructs: `while`, `do-while`, `for`, `break`, `continue` statements
- Functions: declaration, definition and call
- Variable storage class: local variables และ global variables

(2) **Basic concept of testing and debugging** (Chapter 15)

(3) **Pointers and Arrays** (Chapter 16)

- Arrays และStrings
- Pointers
- Passing a reference using pointers และ Passing arrays as function arguments
- ความสัมพันธ์ระหว่าง arrays กับ pointers (ดู Handout 7 ด้วย)
- Problem solving with arrays: Insertion sort และ selection sort
- Multi-Dimensional Array (อาร์เรย์หลายมิติ)

(4) **Recursion** (Chapter 17)

(5) **File Input/Output** (Chapter 18)

- Text files vs. binary files
- C-standard library functions ที่เกี่ยวข้อง file: `fopen`, `fclose`, `fprintf`, `fscanf`, `fputs`, `fgets`, `fputc`, `fgetc`, `fread`, `fwrite`, `fseek`, `FILE`, `EOF`, `NULL`, `SEEK_SET`, `SEEK_CUR`, `SEEK_END`

(6) **Additional C features** (Appendix D)

- C preprocessor: Macro substitution (`#define`)
- Conditional Expression operator: `expA ? expB : expC`
- `typedef`
- Conditional compilation using pre-processing directive (#ifdef, #else, #endif)

(7) **Data structures** (Chapter 19)

- ตัวแปรแบบ structures ในภาษา C
- Arrays of structures และ Pointers and Structures
- Dynamic Memory Allocation: Dynamically sized arrays
- Linked lists

<u>ทบทวน: Linked lists</u>

1. Linked list = โครงสร้างที่ใช้ในการเก็บข้อมูล โดยที่จะประกอบด้วยชุดของ node แต่ละโหนดใช้เก็บข้อมูล 1 element โดยแต่ละโหนดไม่จำเป็นต้องเรียงต่อกันใน memory แต่จะมีฟิลด์ย่อยในโหนดจะเก็บ pointer ชี้ไปยังโหนดที่อยู่ในลำดับที่ต่อกัน (โครงสร้างข้อมูล arrays ข้อมูลแต่ละ element จะเรียงต่อกันใน memory)

2. หลักสำคัญของการเขียนโปรแกรมที่จัดการกับ linked list

นิยามโครงสร้างของ node มี 2 ส่วน คือ ส่วนข้อมูลและส่วน link

```
typedef struct node_tag NODE;
struct node_tag {
    int data;
    NODE *next;
};
int main()
{
int x,y,z;
NODE dummy_head;
NODE *ptr;
  dummy_head.next=NULL
   Add_Node(&dummy_head,x);
   Delete_Node(&dummy_head,y);
   ptr=Find(&dummy_head,z);
}
```

- มี dummy head node ซึ่งไม่ได้เก็บข้อมูลอะไรยกเว้นฟิลด์ next ชี้ไปที่ node แรกของ linked list
- โหนดสุดท้ายจะมีค่าในฟิลด์ next เท่ากับ NULL (ไม่ได้ชี้ไปที่ node ใดๆ)
- operation ที่จำเป็น

(1) เพิ่ม node ใหม่ (ข้อมูลตัวใหม่) ลงใน linked list

```
int Add_Node(NODE *head, int data)
```

(2) ลบ node ใดๆ (เช่น node ทีมี data ที่ต้องการ) ออกจาก linked list

```
int Delete_Node(NODE *head, int data)
```

(3) ค้นหา node ที่มีอยู่ใน linked list (มักจะคืนค่า address ของ node ออกมา)

```
NODE *Find(NODE *head, int data)
```

(4) อื่นๆ แล้วแต่การประบุกต์ใช้

3. การทำความเข้าใจกับ linked list มักจะนิยมวาดโครงสร้างของ linked list ในรูปแบบ abstract form ดังตัวอย่างโจทย์ต่อไปนี้

<u>โจทย์</u> กำหนดให้นิยามของ node ใน linked list เป็นดังนิยามในส่วนย่อยของโปรแกรมข้างล่าง โดยกำหนดให้ค่าหน่วยความจำที่เกี่ยวข้องแสดงดังรูปด้านข้าง โดยที่ฟิลด์ data และ next จะใช้เนื้อที่อย่างละ 1 ที่ตั้ง (location) เรียงต่อกันในหน่วยความจำและกำหนดให้ค่าของอาร์กิวเมนต์ head ในฟังก์ชัน func มีค่าเท่ากับ x800A

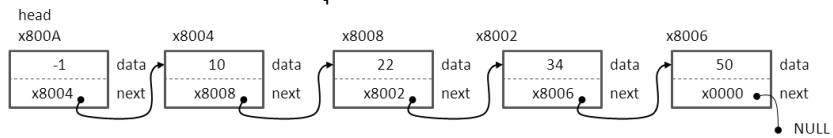
x8000	88
x8001	x0000
x8002	34
x8003	x8006
x8004	10
x8005	x8008
x8006	50
x8007	x0000
x8008	22
x8009	x8002
x800A	-1
x800B	x8004

```
typedef struct node_tag NODE;
struct node_tag {
  int data;
  NODE *next;
};
void func(NODE *head)
{
NODE *new_n,*prev,*curr;
NODE *prev_n,*curr_n,*next_n;
     ...
// Line A_1
    new_n=(NODE *)malloc(sizeof(NODE));
    new_n->data=27;
    new_n->next= prev->next;
    prev->next=new_n;
    prev=curr;
    curr=curr->next;
  printf("%d %d\n",prev->data,curr->data);
// Line A_2
    ...
```

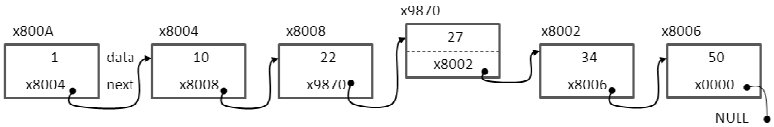
```
  // Line B_1
  prev_n=head;
  curr_n= prev_n->next;
  next_n=curr_n->next;
  next_n=next_n->next;
  next_n=next_n->next;
  while(next_n!=NULL)
  {
       // Line B_2
      printf("%d %d %d\n",prev_n->data,
            curr_n->data,next_n->data);
       prev_n=curr_n;
       curr_n=curr_n->next;
       next_n=next_n->next;
   }
   // Line B_3

}
```

(i) จงวาดรูปแสดงโครงสร้างข้อมูลของ linked list ที่ส่งเข้ามาเป็นอาร์กิวเมนต์ (head) ของฟังก์ชัน func

(ii) สมมติเมื่อฟังก์ชันทำงานถึง Line A_1 แล้ว prev มีค่าเท่ากับ x8008 และ curr มีค่าเท่ากับx8002 และ new_n ถูกกำหนดด้วยค่าที่คืนจาก malloc เท่ากับ x9870 จงวาดรูปแสดงโครงสร้างข้อมูลของ linked list ที่เปลี่ยนไปเมื่อฟังก์ชันทำงานที่ Line A_1 ถึง Line A_2 พร้อมทั้งระบุค่าที่แสดงโดย printf ใน Line A_2



(iii) สมมติว่าฟังก์ชันทำงานต่อที่ Line B_1 ถึง Line B_3 จงระบุค่าต่างๆ ของ prev_n, curr_n และ next_n ที่เปลี่ยนไปในแต่ละ itearation เมื่อโปรแกรมในฟังก์ชัน func ทำงานที่ Line B_2 พร้อมทั้งระบุผลลัพธ์ที่หน้าจอที่ฟังก์ชัน printf แสดงออกสู่หน้าจอใน



รอบที่ 1 prev_n = ……curr_n = ……next_n = ……แสดง……

รอบที่ 2 prev_n = ……curr_n = ……next_n = ……แสดง……

ก่อนหลุดลูป while prev_n = ……curr_n = ……next_n = ……..

(iv) จงเขียนโค้ดในฟังก์ชัน `Add_Node_To_Last` เพื่อทำการเพิ่มข้อมูลโหนดใหม่ลงใน Linked list โดยให้กลายเป็นโหนดสุดท้ายของ linked list

```
void Add_Node_To_Last(NODE *head, int i_data)
{
NODE *ptr, *new_n;

// หาโหนดสุดท้ายใน Linked list (ฟิลด์ next = NULL) -> ptr
   ptr=.....
   while(.....)
   {
      .....
   } ต่อคอลัมน์ถัดไป
```

```
// เพิ่มโหนดใหม่ต่อท้าย โหนดสุดท้าย
……
……
……
……
……
……
}
```

(v) โค้ดในฟังก์ชัน `Free_All_Nodes` ข้างล่างเขียนขึ้นเพื่อคืนข้อมูลในหน่วยความจำที่ได้จองไว้สำหรับแต่ละโหนดใน Linked list แต่ปรากฏว่าโค้ดนี้มี bug อยู่จงอธิบายสาเหตุของบักนี้พร้อมวิธีแก้

```
void Free_All_Nodes(NODE *head)
{
NODE *ptr,*tmp;
      ptr=head->next;
      while(ptr!=NULL)
      {
         free(ptr);      // Line A
         ptr=ptr->next;  // Line B
      }
}
```

สาเหตุของบัก

วิธีแก้

(vi) จงเขียนโค้ดในฟังก์ชัน `Interlace_After` เพื่อทำการแทรกโหนดใหม่ลงไปข้างหลังทุกๆโหนดที่อยู่ใน linked list โดยมีโหนดใหม่ที่แทรกจะมีข้อมูลเท่ากับอาร์กิวเมนต์ `i_data` ตัวอย่าง เช่น ถ้า linked list ปัจจุบันมีทั้งหมด 4 โหนดที่มีข้อมูล 10 45 200 1000 หลังจากเรียกฟังก์ชันโดยส่งอาร์กิวเมนต์ `i_data` เป็นค่า 0 หลังจากฟังก์ชันทำงานเสร็จ linked list จะมีข้อมูลทั้งหมด 8 โหนด ซึ่งมีข้อมูล 10 0 45 0 200 0 1000 0

<u>ทบทวน: Recursion</u>

1. คุณลักษณะของการแก้ปัญหาแบบ recursion คือ

(1) ปัญหาหรือ task หนึ่งๆจะต้องนิยามในรูปแบบของตัวมันเอง โดยที่มีขนาดของปัญหามีขนาดเล็กลง

(2) Task นั้นจะต้องมีนิยามของกรณี Base case ซึ่งจะแก้ปัญหาในรูปแบบ non-recursive (ไม่ได้เรียกตัวมันเอง)

2. ขั้นตอนในการเขียนโปรแกรมเพื่อแก้ปัญหาโดยใช้เทคนิค Recursion

(1) นิยามปัญหาให้อยู่ในรูปของ Recurrence equation (กรณีที่เขียนเป็นสมการได้) หรือให้อยู่ในรูปของ Recurrence relation (ความสัมพันธ์ของปัญหาย่อยที่มีขนาดเล็กลงกับปัญหาใหญ่)

(2) เขียนโค้ดตามนิยามของ Recurrence equation หรือ Recurrence relation ที่ได้จากขั้นตอนที่แล้ว

3. โจทย์ตัวอย่าง จงเขียนโปรแกรมสำหรับหาค่าน้อยสุดในอาร์เรย์โดยใช้เทคนิค Recursion โดยให้ทำตามขั้นตอนต่อไปนี้

(i) ถ้ากำหนดให้นิยามของ Recur_Min(A,n) คือฟังก์ชันที่ทำการหาค่าน้อยสุดในอาร์เรย์ A ที่มีขนาด n ตัว (A[0], A[1], …, A[n-1]) จงเขียน Recurrence equation ของ Recur_Min โดยให้ใช้โอเปอเรเตอร์ min(a,b) ซึ่งหมายถึงค่าที่มากกว่าของ a และ b เช่น min(10,4) มีค่าเท่ากับ 4

(ii) จงเขียนโค้ดภาษาซีเพื่อให้ทำงานตาม Recurrence equation

```
int Recur_Max(int *A, int n)
{
int x;
       if(……)
          ……
       else
       {




       }
}
```

4. โจทย์ตัวอย่าง (ข้อสอบเก่า midterm2 2/2554): จงใช้เทคนิค Recursion เพื่อเขียนโปรแกรมในฟังก์ชัน `SumRange` เพื่อทำการหาผลรวมของข้อมูลอยู่ในอาร์เรย์ A ขนาด n โดยสนใจเฉพาะข้อมูลที่ค่าอยู่ในช่วง lower ถึง upper

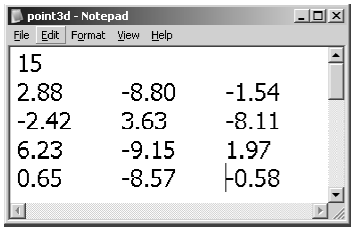
```
int SumRange(int *A, int n, int lower,int upper)
{
```

5. โจทย์ตัวอย่าง (Homework 9): นิยามของ Palindrome คือ สตริงที่มีตัวอักษรจากซ้ายไปขวาเหมือนกับขวาไปซ้ายโดยไม่สนใจ space เช่น "civic", "was it a rat i saw", "11 02 2011" เป็นต้น จงใช้เทคนิค Recursion เพื่อเขียนโปรแกรมทำการตรวจสอบว่าสตริงที่ป้อนเข้ามาเป็น Palindrome หรือไม่ โดยกำหนดให้เขียนโค้ดในฟังก์ชัน IsPal ซึ่งจะทำการตรวจสอบว่าสตริงที่ระบุโดย A ที่มีความยาว N ตัวอักษรนั้นเป็น palindrome หรือไม่ ถ้าใช่ให้คืนค่า 1 ถ้าไม่ใช่ให้คืนค่า 0 โดยกำหนดว่าสตริงจะประกอบด้วยตัวอักษร ‘a’-‘z’, ตัวเลข ‘0’-‘9’ และ Space เท่านั้น

```
int IsPal(char *A,int N)
{
  if(……)
       ……
  if(A[0]==' ' && A[N-1]==' ')
     ……
  else if(A[0]==' ' && A[N-1]!=' ')
     ……
  else if(A[0]!=' ' && A[N-1]==' ')
     ……
  else
       ……
```

<u>ทบทวน: เรื่องอื่น ๆ</u>

1. โจทย์ทบทวนเรื่อง FILE I/O และ structure: กำหนดให้ในไฟล์ point.dat ซึ่งอยู่ใน directory d:\cp111\ เก็บข้อมูลของจุดในปริภูมิ 3 มิติ (3D points) แสดงดังรูปข้างล่าง โดยที่บรรทัดแรกจะระบุจำนวนจุด สมมติคือ N หลังจากนั้น N บรรทัดถัดมาจะเป็นค่า x-, y- และ z-coordinate ของแต่ละจุด หนึ่งจุดต่อหนึ่งบรรทัด



จงเขียนโปรแกรมเพื่อทำการเรียงข้อมูลของเซ็ตของจุดให้เรียงตามค่าระยะห่างระหว่างจุด origin (0,0,0) กับจุดนั้น โดยเรียงจากใกล้สุด (ระยะทางน้อยสุด) ไปยังไกลสุด (ระยทางมากสุด) และให้บันทึกผลลัพธ์ลงไฟล์ sorted.dat โดยเพิ่มข้อมูลระยะทางจากจุดนั้นๆไปจุด (0,0,0) กำหนดให้การเขียนโปรแกรมให้สร้าง Dynamically-sized array of structures โดยให้นิยาม structure ที่เหมาะสม

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
                                        // ใช้ typedef นิยามโครงสร้าง



.......................................................................
void Sort_By_Dist(.................................
void main()
{
.......................................................
FILE *fp;
int i,N;
double x,y,z;
   ..............................................................
   ..............................................................
    {
     printf("cannot open file for reading\n");
     return;
    }
    ..............................................................
   ..............................................................
   {
    printf("cannot allocate memory\n");
    return;
   }
   for(i=0;i<N;i++)
   {
    ..............................................................
   ..............................................................
   ..............................................................
   }
   ..............................................................

    Sort_By_Dist(...............................,N);

    ..............................................................
    ..............................................................
   {
    printf("cannot open file for writing\n");
    return;
   }// -- มีต่อ --
```

```
   // -- ต่อ --

    ..............................................................
   for(i=0;i<N;i++)
   {
        ..............................................................
   }
   ..............................................................

    ..............................................................
}
// ใช้ selection sort เรียง array pts3d ตามระยะทาง

void Sort_By_Dist(.......................* pts3d, int N_pts)
{
...................................  tmp_pts;
double min_d;
int i,j,min_i;

   for(i=0;...................................;.............................)
   {
      // หา ptr3d[min_i] ที่ใกล้กับจุด (0,0,0) ที่สุด
      // ในบรรดา pts3d[i] ... ptr3d[N_pts-1]

       min_d=..............................................................
       min_i=..............................................................

       for(..................................................................)
       {
          if(...................................................................)
          {
                ..............................................................
                ..............................................................
           }
       }
       // สลับ pts3d[i] กับ pts3d[min_i]
       ..............................................................
       ..............................................................
       ..............................................................
    }
}
```

2. โจทย์ทบทวนเรื่อง structure (ดัดแปลงจากข้อสอบ Final 2/2553): กำหนดให้ข้อมูลผลการเรียนวิชาหนึ่งๆ แทนด้วยตัวแปรแบบโครงสร้างชนิด `course_t` ที่นิยามดังข้างล่าง โดยที่ code แทนรหัสวิชา, units แทนจำนวนหน่วยกิตของวิชานั้น และ grade แทนผลการเรียนที่สอบได้ โดยที่ 4, 3.5, 3.0, 2.5, 2.0, 1.5, 1.0 และ 0 แทนคะแนนเกรด A, B+, B, C+, C, D+, D และ E ตามลำดับ

จงเขียนโค้ดในฟังก์ชัน `calculate_GPA` ซึ่งจะคำนวณและคืนค่า GPA ของนิสิตคนหนึ่งที่ลงเรียนในรายวิชาที่ลงทะเบียนเรียนทั้งหมดซึ่งกำหนดโดยอาร์กิวเมนต์ courses ซึ่งจะเป็นแอดเดรสของ element ตัวแรกใน array of structure course_t ที่มีขนาดระบุโดยอาร์กิวเมนต์ `N_courses`

```
typedef struct {
  int code;
  int credits;
  double grade;
} course_t;
```

	code	credits	Grade
courses[0]	101	4	1.5
courses[1]	102	2	0
courses[2]	111	3	2.0
courses[3]	121	1	4.0

```
double calculate_GPA (course_t *courses, int N_courses)
{
```

3. โจทย์ทบทวนเรื่อง FILE I/O: เมื่อโปรแกรมข้างบนทำงานเสร็จ โดยสมมติว่าการเปิดไฟล์โดย fopen ไม่มี error จงระบุข้อมูลที่เก็บอยู่ในไฟล์ d:/cp111/data.txt และผลลัพธ์ที่แสดงออกที่หน้าจอ

หมายเหตุ รหัสแอสกีของ 0 และ 9 มีค่าเท่ากับ 48 ฐานสิบ (หรือ '0') และ 57 ฐานสิบ (หรือ '9') ตามลำดับ

```
#include <stdio.h>
void main()
{
FILE *fp;
int a,b;
char ch;
    fp=fopen("d:/cp111/data.txt","w");
    if(fp==NULL)
        return;
    ch='-';
    fputc(ch,fp);
    a=1;
    ch=a+'0';
    fputc(ch,fp);
    b=a+1;
    fprintf(fp,"%d\n",b);
// -- มีต่อ --
```

```
    b=b+10;
    ch='9';
    fprintf(fp,"-%d %c\n",b,ch);
    fclose(fp);

    fp=fopen("d:/cp111/data.txt","r");
    if(fp==NULL)
       return;
    fscanf(fp,"%d",&a);
    fscanf(fp,"%d\n",&b);
    ch=fgetc(fp);
    fclose(fp);

    printf("%d %d %d\n",a,b,ch);
}
```

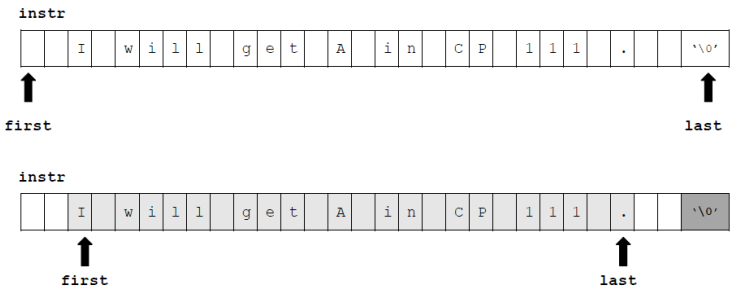
4. โจทย์ทบทวนเรื่อง FILE I/O: เมื่อโปรแกรมข้างล่างทำงานเสร็จสิ้น โดยสมมติว่าการเปิดไฟล์โดย fopen ไม่เกิด error โปรแกรมจะแสดงผลลัพธ์อะไรออกสู่หน้าจอ

```
#include <stdio.h>
void main()
{
FILE *fp;
int x,y,z,i;
int A[10]={4,8,3,6,5,2,1,8,0,7};

   fp=fopen("d:/cp111/ data.bin","wb");
   if(fp==NULL)
      return;

   for(i=0;i<10;i++)
       fwrite(&A[i],sizeof(int),1,fp);

   fclose(fp);
// -- มีต่อ --
```

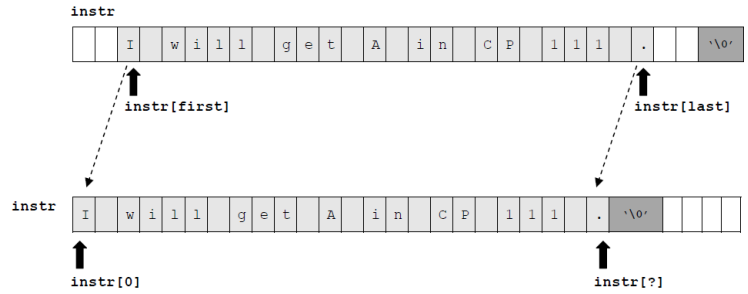
```
   fp=fopen("d:/cp111/ data.bin","rb");
   if(fp==NULL)
      return;
   fread(&x,sizeof(int),1,fp);

   fseek(fp,x*sizeof(int),SEEK_SET);
   fread(&y,sizeof(y),1,fp);

   fseek(fp,2*sizeof(int),SEEK_CUR);
   fread(&z,sizeof(int),1,fp);

   fclose(fp);

   printf("%d %d %d\n",x,y,z);
}
```

5. โจทย์ทบทวนเรื่อง arrays, strings and pointers (ข้อสอบเก่าปี 2552 programming exam #3):: จงเขียนโค้ดในฟังก์ชัน `Trim(char *str)` ซึ่งจะทำการลบตัวอักษร blank space **<u>ที่อยู่ข้างหน้าและข้างหลัง</u>** ออกจากสตริงที่ระบุโดยอาร์กิวเมนต์ `str` ซึ่งมีชนิดข้อมูลเป็น pointer to `char` **<u>โดยที่ blank space ที่อยู่ข้างในจะไม่ถูกลบทิ้ง</u>** ตัวอย่างเช่น ถ้า `str="  CP111 is easy.  "` หลังจากฟังก์ชันทำงานเสร็จ `str="CP111 is easy."`

(1) ให้ไล่สแกนจากต้นสตริงและจากท้ายสตริงเพื่อหาตัวอักษรตัวแรกที่ไม่ใช่ space สมมติว่าแทนด้วย array index **first** และ **last**



(2) copy ตัวอักษรตั้งแต่ element ตัวที่กำหนดโดย **first** จนถึงตัวที่กำหนดโดย **last** ไปยัง element ตัวที่ 0 เป็นต้นไป แล้วใส่ '\0' ต่อท้ายเพื่อทำให้เป็น NULL-terminated string



5. โจทย์ทบทวนเรื่อง pointer notation (ข้อสอบเก่า Mid2 2/2554): จงเขียนโค้ดในฟังก์ชัน `Double_Character` เพื่อทำให้ฟังก์ชันนี้ทำการสร้างสตริง `outstr` จาก `instr` โดยการเพิ่มตัวอักษรใน `instr` ที่ตรงกับตัวอักษรที่ระบุโดย `ch` ขึ้นไปอีก 1 ตัว เช่นถ้า instr คือ "satis" และ ch คือ 's' ดังผลลัพธ์ที่ outstr คือ "ssatiss" นั่นคือตัวอักษร s จะถูกเพิ่ม 1 ตัวในทุกตำแหน่งที่พบใน instr

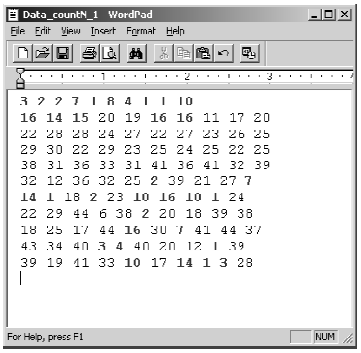
```
void Double_Character(char *instr, char *outstr, char ch)
```

6. โจทย์ทบทวน (ข้อสอบเก่า Mid2 2/2554): จงเขียนโปรแกรมที่วนรับข้อมูลจำนวนเต็มที่มากกว่าหรือเท่ากับ 0 จากคีย์บอร์ดไปเรื่อยๆ จนกว่าจะเจอข้อมูลที่เป็นเลขจำนวนเต็มลบ (จำนวนเต็มลบไม่ถือเป็นส่วนหนึ่งข้อมูล)

<u>หลังจากนั้นให้แสดงจำนวนข้อมูลที่แตกต่างกันที่ถูกป้อนเข้ามา และแสดงผลรวมของข้อมูลเหล่านี้ (จำนวนที่ถูกป้อนซ้ำมากกว่า 1 ครั้งจะถือว่าเป็นข้อมูล 1 ตัวเท่านั้นและจะถูกบวกรวมในผลรวมเพียง 1 ครั้ง)</u>

เช่น ถ้าข้อมูลที่ป้อนเข้ามาคือ 5 1 2 3 5 1 3 -1 ผลลัพธ์ที่แสดงที่หน้าจอคือ 4 11 ซึ่งหมายความว่า 4 จำนวนที่แตกต่างกันที่ถูกป้อนเข้ามาได้แก่ 5 1 2 3 และ ผลรวมเท่ากับ 11=5+1+2+3 นอกจากนี้ให้สมมติว่าข้อมูลที่ป้อนเข้ามาจะไม่เกิน 100 ตัว

7. โจทย์ทบทวน (ข้อสอบเก่าปี 2552 programming exam #3): จงเขียนโปรแกรมที่ค้นหาจำนวนเต็ม 10 จำนวนแรกที่แตกต่างกันที่ถูกเก็บอยู่ไฟล์แบบข้อความ (Text file) เรียงต่อกันไปโดยคั่นด้วย space หรือ `'\n'` (ขึ้นบรรทัดใหม่) ตัวอย่างของข้อมูลในไฟล์สามารถแสดงดังรูปข้างล่างและกำหนดให้ค่าของจำนวนเต็มที่เก็บอยู่ในไฟล์จะอยู่ในช่วง 1-44



นอกจากนี้โปรแกรมจะต้องนับจำนวนครั้ง (occurrences) ที่ตัวเลข 10 จำนวนแรกดังกล่าวปรากฏอยู่ในไฟล์ กำหนดให้โปรแกรมนี้จะต้องรับอินพุตเป็นชื่อไฟล์ข้อมูลจากผู้ใช้ และหลังจากโปรแกรมได้ประมวลผลเสร็จ ให้แสดงจำนวน 10 แรกที่ค้นเจอ พร้อมทั้งจำนวนครั้งที่ปรากฏอยู่ในไฟล์ออกสู่หน้าจอ

8. โจทย์ทบทวน: จงเขียนโค้ดในฟังก์ชัน Find_abc ซึ่งจะทำการค้นหาในสตริงที่ระบุโดยอาร์กิวเมนต์ str เพื่อหา substring ที่อยู่ในรูปแบบ a*b*c* โดยที่ a* หมายถึง ตัวอักษร a อย่างน้อยหนึ่งตัวต่อกัน และเช่นเดียวกับสำหรับความหมายของ b* และ c* โดยฟังก์ชันจะผลิตผลลัพธ์กลับไปที่อาร์กิวเมนต์ first และ last (pointer to int) โดยใช้การส่งค่าแบบ pass by reference ซึ่งเป็นค่า array index ของตัวอักษรตัวแรกและตัวสุดท้ายของ substring ที่พบ ในกรณีที่เจอให้คืนค่า 0 แต่ถ้าไม่เจอให้คืนค่า -1

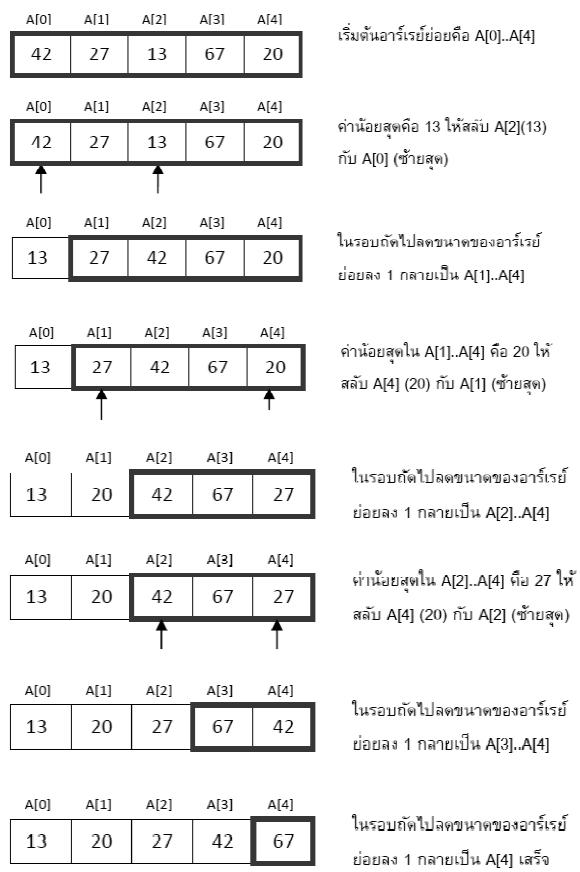
```
int Find_abc(char *str, int *first, int *last)
```

ตัวอย่าง เช่น

- ถ้า str คือ “erceqqaaaabbbcccacsdf” แล้ว substring ที่อยู่ในรูปแบบ a*b*c* คือ “aaaabbbccc” ซึ่งฟังก์ชันจะผลิตค่า first และ last เป็น array index ของตัวอักษรตัวแรกและตัวสุดท้ายของ substring a*b*c* ที่พบใน str ซึ่งในกรณีจะเท่ากับ 6 และ 15 ตามลำดับ และฟังก์ชันคืนค่า 0
- ถ้า str คือ “erceqqaaaacbbbacccacsdf” แล้วจะไม่มี substring ที่อยู่ในรูปแบบ a*b*c* ดังนั้นฟังก์ชันจะคืนค่า -1

9. โจทย์ทบทวน: จงเขียนโปรแกรมสำหรับการสลับตัวอักษรจากหน้าไปท้ายให้เรียงจากท้ายไปหน้า โดยให้เขียนฟังก์ชัน Recur_ReverseStr ซึ่งจะทำการสลับตัวอักษรในสตริง str ที่มีความยาว len ตัวอักษร โดยทำงานแบบ Recursive function และให้เขียนฟังก์ชัน ReverseStr ซึ่งจะทำงานแบบ iteration

```
void Recur_ReverseStr(char *str,int len)
void ReverseStr(char *str)
```

10. โจทย์ทบทวน: จงทำโค้ดในฟังก์ชัน Selection_Sort ให้สมบูรณ์ เพื่อทำให้ฟังก์ชันนี้ทำการเรียงข้อมูลในอาร์เรย์ A ขนาด N element น้อยไป และกำหนดให้ใช้อัลกอริทึมในการเรียงข้อมูลคือ Selection sort ซึ่งอธิบายดังรูปข้างล่าง

```
void Selection_Sort(int *A,int n,int order)
{
int i,j,min_i,min,tmp;

     for(i=0;i<.............;...,,,.........)
     {
       /* find the minimum among A[i] ... A[n-1] */
       min =...............................
       min_i=...............................
       for(j=............;...............;...............)
       {
               if(...............................)
               {
                       ..............................................................
                       ..............................................................
               }
               // Swap A[i] <-> A[m_index]
               ..............................................................
               ..............................................................
       }
     }
}
```