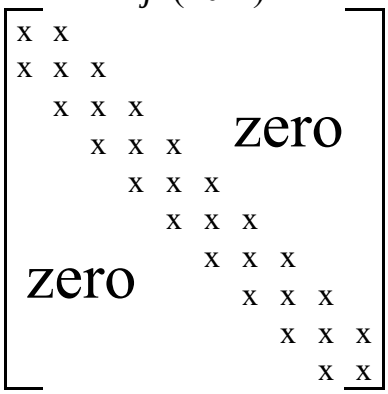
Department of Computer Science and Engineering
National Sun Yat-sen University
Data Structures - Middle Exam, Nov. 14, 2016

1. Explain each of the following terms. (16%)
   (a) private in C++ language
   (b) template in C++ language
   (c) generalized lists
   (d) sparse matrix

2. Calculate the addressing formula for the element $a[i][j][k]$ in an array declared as $a[5][4][7]$. Suppose that the address of $a[0][0][0]$ is 300 and each element requires four bytes. Give the answers for the row-major representation and the column-major representation. (10%)

3. In the following sparse matrix $a[\ ][\ ]$, all elements other than those on the major diagonal and the diagonals immediately above and below this one are zero. Suppose the elements in the band formed by these three diagonals are represented by rows in a linear array b, with $a[0][0]$ being stored in $b[0]$. Suppose that $b[k]$ stores the value of $a[i][j]$, $0 \le i, j \le n-1$. Please calculate the addressing formula for k with i and j. (10%)



4. Explain the meaning of an *equivalence class*. (6%)

5. Transform the *prefix* expression *++A−**BCD/+EF*GHI* to *infix* and *postfix* expressions. Draw its expression tree. (10%)

6. Please present the method to transform an *infix* expression to a *postfix* expression with the help of a stack. And illustrate your method with the example ((A-(B+C))*D)*(E+F). Note that you have to describe your method, but do not write a program. If there is no description of your method, you will get no point. (12%)

7. Write a C/C++ function to perform the selection sort for sorting the input elements into nondecreasing order. (12%)

```
void Sel_Sort(int a[ ], int n)
```

```
// a[ ]: the element array to be sorted.    n: number of elements
{
```

Please write the body of Sel_Sort ( ).

```
} // end of Sel_Sort ( )
```

8. Write a C/C++ function to perform *Push* and *Pop* operations of a stack implemented with an array. (12%)

```
int top;            // top pointer of the stack
int capacity=100;   // size of the stack
char s[100];        // array for the stack
void Push(char x)
   // Push (add) x to the stack. Before the push operation,
   // you have to check if the stack is full.
{
```

(a) Please write the body of Push( ).

```
} // end of Push( )
char Pop( )
// Remove the top element from the stack, and return the removed element.
// Before the pop operation, you have to check if the stack is empty.
{
```

(b) Please write the body of Pop( ).

```
} // end of Pop( )
```

9. Write a C++ function to reverse a singly linked list. For example, suppose that the give list $X=(x_1, x_2, \ldots, x_{n-1}, x_n)$. After the reversing process, the list will become $(x_n, x_{n-1}, \ldots, x_2, x_1)$. (12%)

```
class ChainNode {
   int data;
   ChainNode *link;
};
class Chain {
   ChainNode *first;      // first node of the list
   void Reverse( )    // Reverse the list.
   {
   ChainNode *p, *c;          //    p:previous, c:current
```

Please write the body of Reverse ( ).

```
   } // end of Reverse ( )
};
```

Answer:

1.

(a) private: 用來保護 class 內部的成員(變數與函式)，只有 class 內部成員可使用。

(b) template: class 的樣板，可以用一個代稱來代表資料型態的名稱，之後便可以該樣板來使用不同的資料型態，通常使用在 function 或 class 中，以一個樣板來代替多種資料型態。

(c) generalized lists: 一種 list，每一個 node 的資料內容是一個 atom 或其他 list(或 null list)。

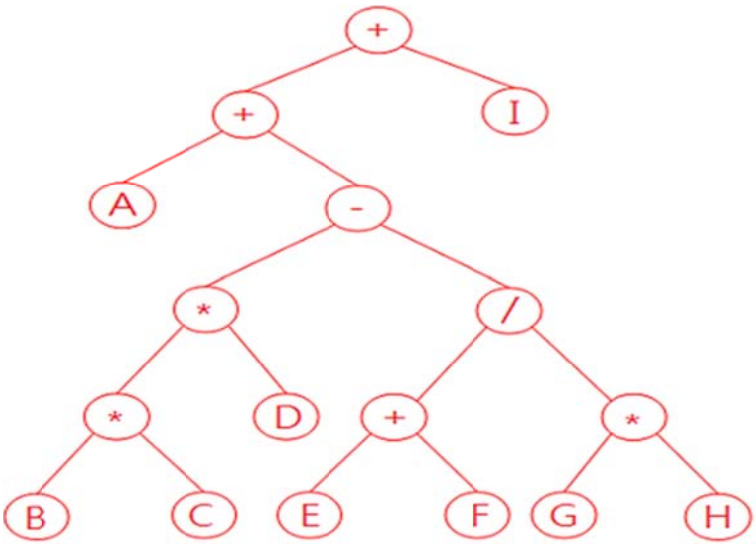
(d) sparse matrix:稀疏矩陣，矩陣內大部分元素為 0，少數儲存具備有意義的值(非零的值)。

2. row-major 300+(i×4×7+j×7+k) ×4
   column-major 300+(i+j×5+k×5×4) ×4

3. k= k=3i+(j-i)=2i+j

4. equivalence class: 等價類，一個集合內的元素可透過反身性、對稱性及遞移性而相互聯結的類別。

5. Infix: A+[B*C*D-(E+F)/(G*H)]+I
   Postfix: ABC*D*EF+GH*/-+I+



6. 轉換方法如下：

(1)遇到數字時，直接 output

(2)遇到左括號或運算符號，則 push 放入到 stack 中

(3)遇到右括號，則把相對應到的左括號上的運算符號 pop 出來

(4)若在放入運算符號的過程中，遇到 stack top 運算符號優先順序較高時，

則把高的優先順序先 pop 出來，再把目前的運算符號 push 到 stack

(5)最後，在將 stack 剩下的運算符號 pop 出來

操作範例如下：

Token	Output	Stack
(		(
(		((
A	A	((
-	A	((-
(	A	((-(
B	AB	((-(
+	AB	((-(+
C	ABC	((-(+
)	ABC+	((-
)	ABC+-	(
*	ABC+-	(*
D	ABC+-D	(*
)	ABC+-D*	
*	ABC+-D*	*
(	ABC+-D*	*(
E	ABC+-D*E	*(
+	ABC+-D*E	*(+
F	ABC+-D*EF	*(+
)	ABC+-D*F+	*
	ABC+-D*EF+*	

最後得到 postfix expression 為 ABC+-D*EF+*

7.

```
void Sel_Sort (int a[ ], int n)
// sort the n integers a[0]~a[n-1] into nondecreasing order
{
      for ( int i = 0; i < n; i++)
      {
```

```
            int j = i; // find the smallest in a[i] to a[n-1]
            for (int k = i+1; k < n; k++)
                  if (a[k] < a[j]) j = k;
            // swap(a[i], a[j]);
            int temp = a[i]; a[i] = a[j]; a[j] = temp;
      }
} // end of Sel_Sort ( )
```

8.

```
void Push (char x)
   // Push (add) x to the stack. Before the push operation,
   // you have to check if the stack is full.
{
      if (top == capacity - 1)
          throw "Stack Overflows";
      s[++top] = x;
} // end of Push( )

char Pop( )
// Remove the top element from the stack, and return the removed element.
// Before the pop operation, you have to check if the stack is empty.
{
    if (top == -1)
        throw "Stack is empty. Cannot delete."
    char x=s[top];
    top--;
    return x;
} // end of Pop( )
```

9.

```
void Reverse( )
        // Reverse the list.
{
   ChainNode *p, *c;          //    p:previous, c:current
   c = first
   p = 0;      // before current
   while (c) {
        ChainNode *r = p;
         p = c;
         c = c ->link;      // moves to next node
         p->link = r; // reverse the link
   }
   first = p;
} // end of Reverse ( )
```