Department of Computer Science and Engineering
National Sun Yat-sen University
Data Structures - Middle Exam, Nov. 17, 2014

1. Explain each of the following terms in C++ language. (12%)
   (a) protected
   (b) constructor
   (c) operator overloading

2. A *upper triangular* array *a* is an *n*-by-*n* array in which *a*[*i*][*j*]==0, if *i*>*j*. Suppose that array *a* is stored in one-dimensional array *b* sequentially with *a*[0][0] being stored in *b*[0]. In other words, the sequence for storing in array *b* is *a*[0][0], *a*[0][1],…, *a*[0][n-1], *a*[1][1], *a*[1][2], …, *a*[1][n-1], *a*[2][2], *a*[2][3],…, *a*[2][n-1], ….Calculate the addressing formula for the element *a*[*i*][*j*] stored in *b*[*k*] in the upper triangular part. (10%)

3. Please give a method, with the help of a stack, to check whether an arithmetic expression containing multiple types of parentheses is valid or not. For example, {x + (y - [a+b]) * c – [(d+e)]} / (h-j) is valid, but [(d+e]) is invalid. (10%)

4. Transform the *prefix* expression *++A−**BCD/+EF*GHI* to *infix* and *postfix* expressions. Draw its expression tree. (9%)

5. What are printed by each of the following C programs? (16%)
   (a)
   ```
   char c=13;    printf("%d \n",~((c >> 3) << 2) );
   ```
   (b)
   ```
   void f(int a[ ], int b[ ], int *c, int *d)
   {   printf("%d %d %d %d \n", a[2],b[2],*(c+2),d[2]);   }
   void main( )
   {   int e[ ]={10,11,12,13,14,15,16,17,18,19,20};
       f(e,e+2,&e[3],&e[3]+2); }
   ```
   (c)
   ```
   int c[ ]={10,11,12,13,14};    int *r;
   r=c+1;        *(r++)=c[0]+5;
   printf("%d %d %d %d \n",c[1],c[2],*r,*(r+1));
   ```
   (d)
   ```
   union {
      char m;
      unsigned char n;
   }u;
   u.n=197;
   printf("%d \n",u.m);
   ```

6. How do you implement the *set* operations (including *intersection, union, difference* and *containment*) in a programming language? Give examples to illustrate your implementation. (10%)

7. The *Fibonacci* sequence is defined recursively as follows:

$$f(n) = n, \quad \text{if } n = 0, 1$$

$$f(n) = f(n-1) + f(n-2), \quad \text{if } n \geq 2.$$

Assume that $f(0)$ and $f(1)$ are given.

(a) Suppose we use an iterative method to compute $f(n)$. How many additions are required? (3%)

(b) Suppose our program is written recursively for computing $f(n)$. How many additions are required? Please derive a general pattern. (6%)

8. Write a recursive C/C++ function to print out all permutations of given elements. (12%)

```
void Permu(char a[ ], int k, int m)
//Generate all the permutations of a[k], ..., a[m]
{
 ┌─────────────────────────────────────────┐
 │ Please write the body of Permu ( ).     │
 └─────────────────────────────────────────┘
} // end of Permu ( )
int main( )
{   char a[ ]={ 'a','b','c','d'};
    Permu(a,0,3);
};
```

9. Let x=(x_1, x_2, …, x_n) and y=(y_1, y_2, …y_m) be two circular chains. Write a C++ function to merge the two circular chains together to obtain the circular chain z=(x_1, y_1, x_2, y_2,…, x_n, y_n, y_{n+1}, …, y_m) , where n ≤ m. (12%)

```
class ChainNode {
   int data;
   ChainNode *link;
};
class Chain {
   ChainNode *first *last;      // circular chain
   Chain merge(Chain &y )
// Merge two circular chains *this (x) and y into a
// single circular chain.
   {
   Chain z;         // The resulting chain
 ┌─────────────────────────────────────────┐
 │ Please write the body of merge ( ).     │
 └─────────────────────────────────────────┘
   return z;
   } // end of merge ( )
};
```