Department of Computer Science and Engineering
National Sun Yat-sen University
Data Structures - Middle Exam, Nov. 19, 2018

1. Suppose an array is declared as *a*[6][5][8], where the address of *a*[0][0][0] is 500 and each element requires four bytes. Please give the formula for calculating the address of *a*[*i*][*j*][*k*] with the row-major and the column-major representations. (8%).

2. What are printed by each of the following C programs? (16%)

   (a)
   ```
   int a[4]={15}, i=1;
   a[++i] += ++i;
   printf("%d %d %d %d\n", a[0], a[1], a[2], a[3]);
   ```

   (b)
   ```
   int b=112;
   printf("%d \n", b & (-b));
   ```

   (c)
   ```
   int c[ ]={20,24,28,32,36};    int *p;
   p=c+2;      *(p++)=c[0]+5;    *(c+1) = *(p+1)+3;
   printf("%d %d %d %d \n",c[1],c[2],c[3],c[4]);
   ```

   (d)
   ```
   union {
      char m;
      int n;
   }u;
   u.n=187;
   printf("%d \n",u.m);
   ```

3. The length of a horizontal wire is 100 cm (the leftmost scale is 0 and the rightmost is 100). There are *n* ants at different starting positions (represented as numbers between 0 and 100). Each ant crawls to the left or right with the same speed 1 cm/sec. Some ants crawl to the left and others crawl to the right. When two ants meet on the way, each crawls back in the opposite direction. When an ant crawls to the left or right end, it will fall down. You are asked to answer the time when the last ant falls down, where the starting time is set to 0 second. For example, *n*=2, the starting positions are (30, right), (60, left), which means that an ant crawls to the right at 30 cm, and an ant crawls to the left at 60 cm. The answer is 70 seconds. Now, *n*=5, the starting positions are (25, left), (30, right), (40, right), (50, left), (75, left). What is the answer? (9%)

4. Please draw the expression tree of the infix expression ((A-(B+C))*D)*(E+F), and then give the prefix and postfix forms. (9%)

5. Please present the method for converting an infix expression into its postfix expression with the help of a stack. You may use the example of the above problem to explain your method. But, you should explain the method clearly, not only demonstrate an example. (12%)

6. Explain each of the following terms. (12%)
   (a) template in C++ language
   (b) circular queue
   (c) generalized list

7. The *Fibonacci-like* sequence is defined recursively as follows:
   $f(n) = n$, if $n = 0, 1, 2$
   $f(n) = f(n-3) + f(n-2) + f(n-1)$, if $n \geq 3$.
   Assume that $f(0)$, $f(1)$ and $f(2)$ are given.
   (a) Suppose we use an iterative method to compute $f(n)$. How many additions are required? (5%)
   (b) Suppose our program is written recursively for computing $f(n)$. How many additions are required? Please derive a general formula. (5%)

8. Write a recursive C/C++ function to solve the Hanoi tower problem for moving n disks from peg A to peg C with auxiliary peg B, $n \geq 1$. For each disk movement, you have to print the message such as "move disk 3 from peg B to peg A". Note that a larger disk must be put below a smaller disk at any time. (12%)

```
void tower(….)
// the recursive function, printing the movement message
{
Please write the body of tower( ).
} // end of tower( )
void main( )
{
   int n=10;
   tower(n, 'A', 'B', 'C');
} /* end of main ( )*/
```

9. Write a C++ function to reverse a singly linked list. For example, suppose that the given 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. row-major $500+(i\times40+j\times8+k)\times4$
   column-major $500+(i+j\times6+k\times30)\times4$
2. (a) 15 0 0 2 (b) 16 (c) 39 25 32 36 (d) -69
3. 75
4. Prefix:**-A+BCD+EF
   Postfix: ABC+-D*EF+*

5.

6. (a) 2(n-2), n>=3 (b) g(n)= g(n-3)+g(n-2)+g(n-1)+2, n>=3, g(0)=g(1)=g(2)0

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 ( )
```