Department of Computer Science and Engineering
National Sun Yat-sen University
Data Structures - Middle Exam, Oct.24, 2022

1. What are printed by each of the following C programs? (20%)

(a)
```
char x[ ]= "ABCDE", y[ ]= "067285";
printf("%d %d %d %d \n", x[4]-x[1], x[5]-x[1], y[5]-y[1],y[6]-y[3]);
```

(b)
```
int a[ ]={20,21,22,23,24,25};
int n=10;
for (int i=0; i < 4;)
   if (i%2 ==0)
      a[i++] = ++n;
   else
      a[++i] = n++;
printf("%d %d %d %d \n",a[0], a[1], a[3], a[4]);
```

(c)
```
int x=28, y=18;
printf("%d   %d\n", x&(x-1), y&(y-1)); // &: bitwise AND
```

(d)
```
void f(int a[ ], int b[ ], int *x, int *d)
{   printf("%d   %d   %d   %d \n", a[0],b[4],*(x+2),d[3]);   }
int main( )
{   int c[ ]={10,11,12,13,14,15,16,17,18,19,20};
    f(&c[0]+2, c+3, &c[3] ,&c[2]+3); }
```

(e)
```
int a=26, b=14, c=5;
printf("%d %d \n",a^b^a^b^b, (b^a^c^a^b^c^a^b^b^c)+b );   //^:XOR
```

2. Determine the frequency count for the statement in line iv of the following program segment: (10%)

```
 i.   for (i=1; i<=n; i++)
 ii.     for (j=1; j<=i; j++)
 iii.       for (k=1; k<=j; k++)
 iv.           x++;
```

3. An array is declared as $a[4][5]$. Now we have an element $a[x][y]$. Suppose row-major addressing and column-major addressing are used. What are the values of x and y so that $a[x][y]$ locates at the same address when the two addressing methods are used? (10%)

4. Given an postfix expression AB-C*D-EF+/G-, please draw its expression tree, and then give the infix and prefix forms. (10%)

5. Suppose A=1, B=1, C=2 and D=3. The value of a list with some of these symbols is the sum of the used symbols. For example, the total values of some permutations are shown: A=1, B=1, AB=2, BA=2, CDD=8, DCD=8, BCDC=8, ABCDA=8. The value of a list is represented as n. Let $f(n)$ denote the number of lists whose values

are n. Then $f(1)=2$, since there are 2 permutations with value 1: A, B. $f(2)=5$ is obtained from the 5 lists: AA, BA, AB, BB, C. $f(3)=13$ is obtained from the 13 lists: AAA, BAA, ABA, BBA, CA, AAB, BAB, ABB, BBB, CB, AC, BC, D. In addition, we set $f(0)=1$ for initialization. Please derive the recurrence formula for calculating $f(n)$, $n \geq 4$. (10%)

6. Explain each of the following terms. (16%)
   (a) inheritance in C++ language
   (b) circular queue
   (c) equivalence relation
   (d) Generalized lists

7. Write a recursive C/C++ function to perform the binary search on a sorted array. (12%)

```
… Binary(….)
// Binary search on a sorted array. Please define the parameters for the function
and the returned value
{
```

Please write the body of Binary ( ).

```
} // end of Binary ( )
```

8. 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 {
  public:
    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 ( )
};    // end of class Chain
```

Answers:

1. (a) 3 -66 -1 -50
   (b) 11 21 13 13
   (c) 24 16
   (d) 12 17 15 18
   (e) 14 45

Explanation:

(a) x[0]='A'=65, x[1]='B'=66, x[2]='C'=67, x[3]='D'=68, x[4]='E'=69, x[5]='\0'=0.
Thus, x[4]-x[1]=3, x[5]-x[1] = -66,
y[0]='0'=48, y[1]='6'=54, y[2]='7'=55, y[3]='2'=50, y[4]='8'=56, y[5]='5'=53, y[6]='\0'=0.
Thus, y[5]-y[1]= -1, y[6]-y[3] = -50

(b)

```
int a[ ]={20,21,22,23,24,25};
int n=10;
for (int i=0; i < 4;)
    if (i%2 ==0)
        a[i++] = ++n;
    else
        a[++i] = n++;
```

We have:
when i=0, a[i++] = ++n, thus a[0]=11; n=11
when i=1, a[++i] = n++, thus a[2]=11; n=12 a[1]=21;
when i=2, a[i++] = ++n, thus a[2]=13; n=13
when i=3, a[++i] = n++, thus a[4]=13; n=14 a[3]=23;

(c) Suppose the 8-bit space is used
x=28=00011100; x-1= 00011011 x&(x-1) = 00011000 =24
y=18=00010010; y-1= 00010001 y&(y-1) = 00010000 =16
In fact, x&(x-1) is to convert the rightmost bit with one to zero.

(d)

a[0] corresponds to c[2];
b[0] corresponds to c[3];
x[0] corresponds to c[3];
d[0] corresponds to c[5];

(e)

a^b^a^b^b=6=14; //^:XOR
(b^a^c^a^b^c^a^b^b^c)+b
=(a^c)+b=(26^5)+b=(00011010^00000101)+b
=00011111+b=31+14=45

2.

$$1 + (1+2) + (1+2+3) + (1+2+3+4) + \ldots + (1+2+3+\ldots+n)$$

$$= \sum_{i=1}^{n} (1 + 2 + \cdots + i)$$

$$= \sum_{i=1}^{n} \frac{i(i+1)}{2}$$

$$= \frac{1}{2}\left(\frac{n(n+1)(2n+1)}{6} + \frac{n(n+1)}{2}\right)$$

$$= \frac{n^3 + 3n^2 + 2n}{6} = \frac{n(n+1)(n+2)}{6}$$

3.

Answer: x=3, y-4.

Explanation:

Row-major addressing: 5x+y

Column-major addressing: 4y+x

We have 5x+y=4y=x

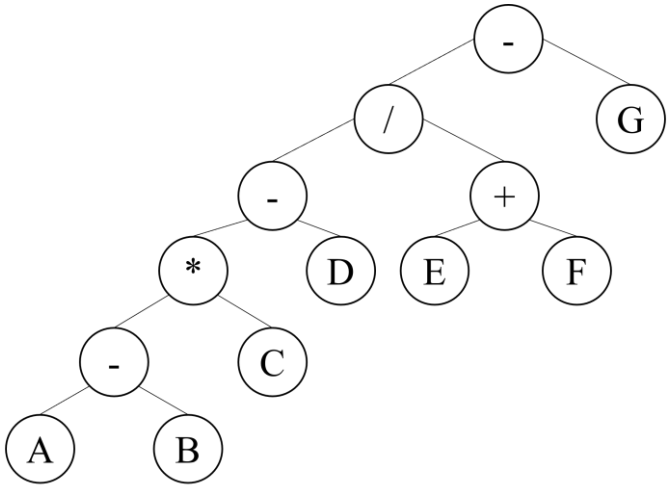
Then 4x=3y. Thus x=3, y-4.

4.

postfix: AB-C*D-EF+/G-

infix: ((A-B)*C-D)/(E+F)-G

prefix: -/-*-ABCD+EFG



5.

f(n) = 2f(n-1) + f(n-2) + f(n-3)

2f(n-1): append A or B to each of f(n-1) with A or B, then the value is increased by 1 (from n-1 to n).

f(n-2): append C to each of f(n-2), then the value is increased by 2 (from n-2 to n).

f(n-3): append D to each of f(n-3), then the value is increased by 3 (from n-3 to n).

6.

(a) inheritance in C++ language: a feature or a process in which, new classes are created from the existing classes.

(b) circular queue: in order to avoid the problem that the queue is not full, but new data items cannot be inserted, the indicators at the head and tail of the queue can be made to wrap around to the beginning of the array.

(c) equivalence relation: a kind of binary relation that should be reflexive, symmetric and transitive

(d) Generalized Lists: a finite sequence of n elements ($n \geq 0$). The element e_i is either an atom (single element) or another generalized list.

7.

```
int Binary (int *a, int x, const int left, const int right)
//Search a[left], ..., a[right] for x
{
   if (left <= right) {
      int middle = (left + right)/2;
      if (x < a[middle])
         return Binary (a, x, left, middle-1);
      else if (x > a[middle])
         return Binary (a, x, middle+1, right);
      else return middle;
   } // end of if
   return -1;// not found
} // end of Binary
```

8.

```
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 ( )
```