以下 6 個の設問より、5 個を選んで回答せよ。

1　再帰処理の動作予測 (× 20)

```
int max( int a[] , int l , int r ) {
    int ans ;
    if ( r - l == 1 ) {
        ans = a[ l ] ;
    } else {
        int m = (l + r) / 2 ;
        int lm = max( a , l , m ) ;
        int rm = max( a , m , r ) ;
        if ( lm > rm )
            ans = lm ;
        else
            ans = rm ;
    }
    printf( "%d\n" , ans ) ; /* (A) */
    return ans ;
}
int a[] = {
    44 , 22 , 33 , 11
} ;
void main() {
    printf( "%d\n" , max( a , 0 , 4 ) ) ;
}
```

設問. 左に記載した再帰呼出しのプログラムを実行した時、(A) によって表示される内容を答えよ。(20)

2　再帰方程式と一般式 (× 20)

前節問の max() のデータ件数 N に対する処理時間 $T_{max}(N)$ を分析したい。

1. この処理の処理時間に相応しい再帰方程式を示せ。
   式の中に使用する時定数等などは、その意味を説明せよ。(10)

2. 再帰方程式を解いて、処理時間の一般式を示せ。
   厳密な証明は不要だが、一般式の導出の流れが分かるように説明せよ。(10)

3　リスト処理の入り口 (20)

リスト構造の理解のために、以下の様なdata と、次のデータの配列の添字番号next からなる配列とした。データ列は必ず 0 番目から始まり、末尾の添字番号は -1 とする。以下の例では、12,23,34,45,56 と昇順に並んでいる。

このデータ列が昇順に並んでいるか判定するプログラムを記述せよ。
昇順で 1 を返し、昇順でない場合は 0 を返すこと。

```
struct LIST {           |   int ascending( struct LIST [] ) {
   int  data ;          |
   int  next ;          |
} table[ 5 ] = {        |
   { 12 , 2 } , // 0    |
   { 34 , 4 } , // 1    |
   { 23 , 1 } , // 2    |
   { 56 ,-1 } , // 3    |
   { 45 , 3 } , // 4    |
} ;                     |
```

4　処理時間とオーダ (20)

1. アルゴリズム A は処理時間が $O(N \log N)$ となるプログラムがある。データ件数 $N = 100$ で 100[msec] であった場合、データ件数が $N = 1000$ では、どの程度の処理時間を要するか答えよ。(10)

2. データ件数が N で表される、プログラムの処理時間 $T(N)$ が、以下の一般式で与えられる場合、処理時間をオーダ記法で示せ。式中の $T_\alpha, T_\beta, T_\gamma$ は、一定時間とする。(10)

   ヒント: $(a^x)' = a^x \log a$

   $T(N) = T_\alpha + T_\beta N\sqrt{N} + T_\gamma 1.1^N$

5 説明問題 (20)

1. アルゴリズムの処理時間が $O(N^2)$ となるプログラムがある。与えられた処理時間で、コンピュータ A では、このプログラムは N 件のデータを処理できた。同じプログラムを処理速度 2 倍のコンピュータ B で実行した場合、A と比べて何倍のデータ件数を処理できるか説明せよ。(10)
2. 処理速度・メモリの使用量・プログラムの複雑さの間の関係を、具体例を交えながら説明せよ。(10)

6 プログラム記述 (20)

複数の頂点の XY 座標を VPSIZE 個読み込んで、表示するプログラムを作成する。入力データは、頂点数と頂点数の XY 座標がVPSIZE 個とする。

```
#define VPSIZE 3          | 入力例
struct VecPoints {        |  3  10 10  20 20  10 20
  int  size ;             |  4   0  0   0 10  10 10  10  0
  int* points ;           |  2  45 45  50 50
} ;                       +-------------------------------
void main() {
   struct VecPoints* vp ;
   vp = (struct VecPoints*)malloc( sizeof( struct VecPoints ) * VPSIZE ) ;

   if (                 ) {
        ~~~~~~~~~~~~~~~~(A)3
      int i , j ;
      for( i = 0 ; i < VPSIZE ; i++ ) { // 座標データを読み込む

         scanf( "%d" ,                        ) ; // 頂点数を読む
                       ~~~~~~~~~~~~~~~~~~~~~~~(B)3
         vp[ i ].points =                                                          ;
                          ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(C)3
         if (                            )
              ~~~~~~~~~~~~~~~~~~~~~~~~~~~(D)3
            exit( 1 ) ;
         for( j = 0 ; j < vp[ i ].size ; j++ ) // 頂点数だけの XY 座標を読む

            scanf( "%d%d" ,                                                          ) ;
      }                     ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(E)3
      for( i = 0 ; i < VPSIZE ; i++ ) { // すべての座標を出力する
         for( j = 0 ; j < vp[ i ].size ; j++ )
            printf( "(%d,%d) " , vp[ i ].points[ 2*j ] , vp[ i ].points[ 2*j + 1 ] ) ;
         printf( "\n" ) ;
      }
      for( i = 0 ; i < VPSIZE ; i++ )

         ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(F)3 ; // XY 座標の配列を捨てる
                                               // 全体の配列を捨てる
   }  ~~~~~~~~~~~~~~~~~~~(G)2
}
```