情報構造論 (斉藤)　後期中間試験　　番号：

4EI　2015/11/27　(1/3)　　氏名：

以下の 5 つの設問の中から、4 つを選んで解答せよ。

1　ヒープのトレースと穴埋め (25)

```
int array[7] = { 55 , 73 , 26 , 81 , 61 , 41 , 14 } ;

int find( int a[] , int n , int size , int key ) {
   while( n < size ) {
      printf( "%d " , n ) ;
      ~~~~~~~~~~~~~~~~~~~~~(A)
      if ( a[ n ] == key )
         return key ;
      else if ( a[ n ] > key )
         n = 2 * n + 2 ;
      else
         n = 2 * n + 1 ;
   }
   return -1 ;
}
```

設問 1 では、この表示命令の結果も書くこと。

```
void main() {
    printf( "%d\n" ,
            find( array , 0 , 7 , 81 ) ) ;
    printf( "%d\n" ,
            find( array , 0 , 7 ,  1 ) ) ;
}
```

```
void print_all( int a[] , int n , int size ) {

   if ( _________________ ) {
        ~~~~~~~~~~~~~~~~(A)

      print_all( a ,                  , size ) ;
                   ~~~~~~~~~~~~~~~~(B)
      printf( "%d " , a[ n ] ) ;

      print_all( a ,                  , size ) ;
   }                ~~~~~~~~~~~~~~~~(C)
}
```

設問 1　上記 main を実行した時、表示される内容を答えよ。(13)

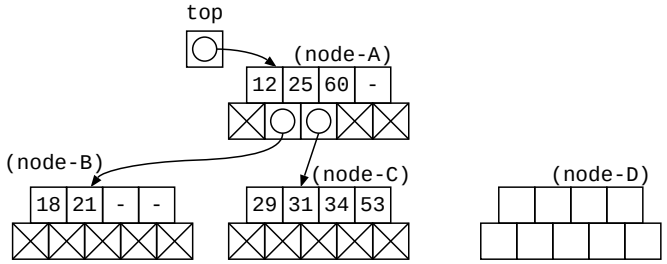
設問 2　配列の中身全部を降順に表示する関数 print_all() の下線部を埋めよ。(4x3)

```
void main() { //呼出の例
  print_all( array , 0 , 7 ) ;
}
```

2　B 木の説明 (25)

以下に示すような位数 2 の B 木がある。このデータのイメージ図を用いて、以下の点を説明せよ。

1. 数値 21 を探す場合の処理の流れを、B 木の特徴を交えて説明せよ。(15)
2. この B 木に数値 30 を追加する場合、どのようにデータが追加されるか (node-A)～(node-D) の欄がどう変化するか判るように、図示せよ。(10)



3　チェイン法 (25)

```
struct NPList {                    | #define HASH_SIZE 10
   char           name[ 10 ] ;     | struct NPList* table[ HASH_SIZE ] ;
   int            phone ;          |
   struct NPList* next ;           | int hash_func( int ph ) {
} ;                                |    return ph % HASH_SIZE ;
                                   | }

struct NPList* npcons( char s[] , int p , struct NPList* x ) {
   struct NPList*n ;
   n = (struct NPList*)malloc( sizeof( struct NPList ) ) ;
   if ( n != NULL ) {                  ___________________________
      strcpy( n->name , s ) ;         |
      n->phone = p ;                  | void entry( char s[] , int ph ) {
      n->next  = x ;                  |    int idx = hash_func( ph ) ;
   }                                  |    table[ idx ] =
   return n ;                         |       npcons( s , ph , table[ idx ] ) ;
}                                     | }

                                        ___________________________
void main() {                          |
   entry( "t-saitoh" , 123456 ) ;      | 設問 1 プログラムの (A) 時点で
   entry( "tomoko" ,   112233 ) ;      |      生成されたデータ構造の
   entry( "mitsuki" ,  222666 ) ;      |      イメージ図を示せ。(15)
   /* (A) */                             ___________________________
   printf( "%s\n" , find( 112233 ) ) ;  |
}                   ~~~~~~~~~~~~~~(B)   | 設問 2 (B) の find で検索し、
                                        |      名前が表示できるように
          find(          ph ) {         |      プログラム中の下線部
~~~~~~~~~(C)    ~~~~~~~~(D)            |      を埋めよ。(2x5)
   int idx =                          ;
             ~~~~~~~~~~~~~~~~~~~~~~~~~(E)
   struct NPList* p ;

   for(                       ; p != NULL ;             ) {
       ~~~~~~~~~~~~~~~~~~~~(F)            ~~~~~~~~~~~~(G)
      if ( p->phone == ph )
         return p->name ;
   }
   return NULL ;
}
```

4　説明問題 (25)

以下の説明問題に答えよ。

1. 逆ポーランド記法について説明せよ。(10)
2. 意志決定木について説明し、そのデータを保存するのにふさわしいデータ構造の宣言を示せ。(10+5)

5　2 分木のプログラム (25)

```
struct Tree {                 | struct Tree* tcons( int x , struct Tree* l , struct Tree* r ) {
   int          data ;        |    struct Tree* n ;
   struct Tree* left ;        |    n = (struct Tree*)malloc( sizeof( struct Tree ) ) ;
   struct Tree* right ;       |    if ( n != NULL ) {
} ;                           |        n->data = x ; n->left = l ; n->right = r ;
                              |    }
struct Tree* top = NULL ;     |    return n ;
                              | }
void additem1( int x ) {
   struct Tree** ppt = &top ;

   while(            != NULL ) {
         ~~~~~~~~~~(A)
      if ( (*ppt)->data == x )
           ~~~~~~~~~~~~(B)
         break ;
      else if (                    )
               ~~~~~~~~~~~~~~~~~~(C)
         ppt =                     ;
      else     ~~~~~~~~~~~~~~~~~~~(D)
         ppt = &((*ppt)->right) ;
   }           ~~~~~~~~~~~~~~~~(E)
   if ( (*ppt) == NULL )
       ~~~~~~(F)
      (*ppt) =                              ;
}             ~~~~~~~~~~~~~~~~~~~~~~~~~~~(G)

struct Tree* additem2( struct Tree* p , int x ) {
   if ( p == NULL ) {
      return tcons( x , NULL , NULL ) ;
   } else {
      if ( p->data != x ) {
         if ( p->data > x )
            p->left  = additem2( p->left , x ) ;
         else
            p->right = additem2( p->right , x ) ;
      }
      return p ;
   }
}
```

設問 1　2 分木にデータを追記する処理の
(A,C,D,G) は処理を埋め、 (3x4)
(B,E,F) はその型を答えよ。(2x3)

(B)_____

(E)_____

(F)_____

```
void main() {
   additem1( 12 ) ;
   additem1(  5 ) ;
   additem1( 20 ) ;
   // この時点で下図イメージが
   // 生成されるように。
   additem2( 14 ) ;
   // <<設問 2>>
}
```

修正
top = additem2(top , 14) ;

設問 2　上記のように ~~additem2(14)~~
を実行したら、どのような
データ構造となるか答えよ。
(7)