情報構造論 (斉藤)　　学年末試験　　番号：

4EI　　2019/2/5(火)　　(1/3)　　氏名：

5 つの選択問題より、4 つを選んで回答せよ。

1　処理時間予測 (25)

以下の設問に、処理時間のオーダーなどの説明を交えながら、処理時間について答えよ。

1. 左右均等な 2 分探索木のデータから特定のデータを探す処理で、$N = 100$ 件の時に 1[μ sec] かかった。データ件数 $N = 1000$ の場合、何秒かかるか答えよ。(9)

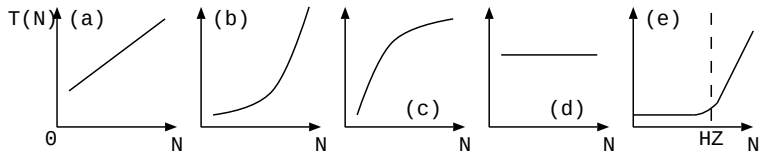
2. 以下の処理時間が、データ件数に対しどのように変化するのか図よりふさわしいものを選べ (4x4)

   (I) チェイン法で新しくデータを追加するのにかかる時間

   (II) B 木で目的のデータを探すのにかかる時間

   (III) クイックソートで配列をソートするのにかかる時間

   (IV) 単純リストの先頭にデータを挿入するのにかかる時間



2　説明問題 (25)

B 木に対しデータを追加する場合の処理について、(a) 位数,(b) ノードの構造や条件,(c) ノードが溢れた時の処理などを交えながら説明せよ。

3　プログラムの動作トレース (25)

```
struct ListRC {
   int refc ;
   int data ;
   struct ListRC* next ;
} ;

struct ListRC* cons( int x , struct ListRC* n )
{
   struct ListRC* ans = (struct ListRC*)malloc( sizeof( struct ListRC ) ) ;
   if ( ans != NULL ) {
      ans->refc = 1 ;
      ans->data = x ;
      ans->next = n ;
   }
   return ans ;
}

struct ListRC* copy( struct ListRC* ans ) {
   if ( ans != NULL )
      ans->refc++ ;
   return ans ;
}

void free_list( struct ListRC* p ) {
   if ( p != NULL )
      if ( --(p->refc) == 0 ) {
         printf( "(%d)%d\n" , p->refc , p->data ) ;
         free_list( p->next ) ;
         free( p ) ;
      }
}

void print_list( struct ListRC* p ) {
  for( ; p != NULL ; p = p->next )
    printf( "(%d)%d " , p->refc , p->data ) ;
  printf( "\n" ) ;
}

void main() {
   struct ListRC* top1 = cons( 11 , cons( 22 , cons( 33 , NULL ) ) ) ;
   struct ListRC* top2 = cons( 44 , copy( top1->next->next ) ) ;
   print_list( top1 ) ;
   print_list( top2 ) ;
   free_list( top1 ) ;
   free_list( top2 ) ;
}
```

【設問】
このプログラムの実行結果を答えよ。
データ構造のイメージ図が正しく記載されていれば、
回答が間違っても中間点を与える。

4　説明問題 (25)

以下の中から 2 つを選択して図を交えて具体的に回答せよ。(12+13)

1. ガベージコレクタについて説明せよ。
2. チェイン法について説明せよ。
3. 動的メモリのヒープについて、確保するメモリサイズが一定の場合で、説明せよ。

5　穴埋め問題 (25)

```
#define HASH_SIZE 26
struct NamePhone {

      name          ;
   ~~~~~~ ~~~~~~~~~~~~(A)
   int   phone ;
} ;
// すべて NULL で初期化
struct NamePhone* table[ HASH_SIZE ] ;

// ハッシュ関数は、1 文字目が'a' の時 0 となる関数
int hash_func( char n[] ) {
                                                  
   return (                ) % HASH_SIZE ;
}          ~~~~~~~~~~~~~~~~(B)
                                                  
void entry( char n[] , int ph ) {
                                                  
   int idx =                                ;
             ~~~~~~~~~~~~~~~~~~~~~~~~~~~~(C)
   while(                                   )
          ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(D)
      idx = (idx + 1) % HASH_SIZE ;
                                                  
   if ( table[ idx ] == NULL ) {
      table[ idx ] = (struct NamePhone*)malloc(                              ) ;
      if ( table[ idx ] != NULL ) {                ~~~~~~~~~~~~~~~~~~~~~~~~~~(E)
         strcpy( table[ idx ]->name , n ) ;

                                    = ph ;
      }  ~~~~~~~~~~~~~~~~~~~~~~~~~~~(F)
   }
}
int search( char n[] ) { // 名前から電話番号を返す

   int idx =                                ;
             ~~~~~~~~~~~~~~~~~~~~~~~~~~~~(G)
   while( table[ idx ] != NULL ) {
      if ( strcmp( table[ idx ]->name , n ) == 0 )
         return table[ idx ]->phone ;

      idx =
  }          ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(H)
  return -1 ;
}
```

【設問】
オープンアドレス法で、名前と電話番号のデータベースを作り、名前から電話番号を探すプログラムを作成する。
コメントの指示に従い穴埋めせよ。

```
void main() {
   entry( "aha" ,     621111 ) ;
   entry( "beatles" , 123456 ) ;
   entry( "apple" ,   765432 ) ;
   printf( "%d\n" ,
           search( "apple" ) ) ;
}
```