情報構造論 (斉藤)　学年末試験　番号：

4EI　2020/2/7(金)　(1/3)　氏名：

5 つの選択問題より、4 つを選んで回答せよ。

1　処理時間の見積もり (25)

1. 以下の処理を行う場合の、処理に要する時間について、横軸 N データ件数、縦軸時間 $T(N)$ にて (a) オーダー記法による処理時間と (b) グラフの概形を描いて答えよ。(5x3)

   (a) B 木に保存されているデータから、特定のキーのデータを探す処理。

   (b) 配列に昇順に保存されているデータに、1 件のデータを挿入する処理。

   (c) オープンアドレス法で保存されているデータに、1 件のデータを加える処理。ただし、N はハッシュサイズより十分小さいものとする。

2. クイックソートで、配列の中身を並び替える処理で、$N = 1000$ 件の時に 3 [msec] かかった。$N = 1,000,000$ 件 では、何秒かかるか答えよ。(10)

2　説明問題

以下の 3 つの説明問題から 2 つを選んで回答せよ。(12+13)

1. 参照カウンタ法について説明せよ。
2. オブジェクト指向プログラミングを取り入れる利点について説明せよ。
3. ハッシュ関数に求められる特徴について説明せよ。

3　フリーリストとイメージ図 (25)

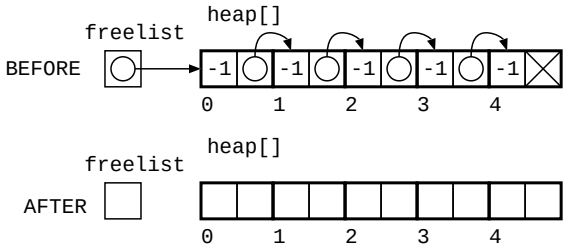
```
struct List {
   int          data ;
   struct List* next ;
} ;

struct List* my_malloc( int x ) {
   struct List* ans = freelist ;
   if ( ans != NULL ) {
      freelist = freelist->next ;
      ans->data = x ;
   }
   return ans ;
}

void main() {
  struct List* a = my_malloc( 3 ) ;
  struct List* b = my_malloc( 4 ) ;
  struct List* c = my_malloc( 5 ) ;
  my_free( a ) ;
  my_free( b ) ;
  my_free( c ) ; // (A)
}
```

```
| struct List  heap[ 5 ] ;
| struct List* freelist = heap ;
|
|
| void my_free( struct List* p ) {
|    p->next = freelist ;
|    freelist = p ;
| }
|_____________________________________
```

(設問)
malloc の動作を理解するための
このプログラムで、配列 heap は、
図 BEFORE のように初期化されている。
処理 (A) が終わった時点でのメモリの
状態のイメージ図を、図 AFTER に
記載せよ。



4　データ構造の設計

買い物サイトを構築するために、以下の様なクレジットカードのデータを大量 (最大 1 万件程度とする) に管理したい。

カード番号	名前	有効期限	セキュリティコード
4340-1122-3344-5678	TOHRU SAITOH	2023 年 09 月	432
4980-0101-2345-6789	AIKO KAORI	2020 年 11 月	729

ただし、カード番号の先頭 4 桁 (例:4980=VISA) は、カード会社の識別番号。

1. C 言語にて、このようなデータを覚えるために相応しいデータ構造を宣言し、(8)
2. 与えられたデータをどのように格納し、どのように検索するのか説明し、(10)
3. クレジットカードの番号からそのデータを探す処理時間のオーダを答えよ。(7)

ただし、配列を先頭から探すような処理の場合は、点数を半分とする。

5　チェイン法穴埋め (25)

IPv4 アドレス (192.156.145.1) とそのドメイン名 (www.fukui-nct.ac.jp) といった情報をチェイン法で管理する、以下のプログラムで作成した。下線部に適切な処理を記載せよ。

```
struct IpHostList {
   int  ip[ 4 ] ;
   char host[ 256 ] ;
   struct IpHostList* next ;
} ;

struct IpHostList* table[ 256 ] ;  // ポインタは全て NULL で初期化

struct IpHostList* ihl_cons( int ad[4] , char hn[256] , struct IpHostList* nx ) {
   struct IpHostList* ans =

      ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(A)4
   if ( ans != NULL ) {
      for( int i = 0 ; i < 4 ; i++ ) {

      }  ~~~~~~~~~~~~~~~~~~~~~~~~~~(B)3
      strcpy( ans->host , hn ) ;
      ans->next = nx ;
   }
   return ans ;
}

char* dig_x( char* sad ) {
   int ad[4] , idx ;

   str_ip4( ad , sad ) ;

   idx =
         ~~~~~~~~~~~~~~~~~~~~(E)3
   for( struct IpHostList* p =                    ;
        p != NULL ;             ~~~~~~~~~~~~~~~~~~(F)3
        ~~~~~~~~~~~~~~~~~) {
                          (G)3
      int flag = 1 ;
      for( int i = 0 ; i < 4 && flag ; i++ ) {

         if (                              )
             ~~~~~~~~~~~~~~~~~~~~~~~~~~~~(H)3
            flag = 0 ; // 違うアドレスだった
      }
      if ( flag )
         return p->host ;
   }
   return NULL ;
}
```

```
// 文字列の IP アドレスを int[4] に変換
void str_ip4( int ad[4] , char* sip ) {
   sscanf( sip , "%d.%d.%d.%d" ,
           &ad[0] , &ad[1] , &ad[2] , &ad[3] ) ;
}

      hash(              ad         ) {
~~~~~~~(C)3   ~~~~~~~~~~~~~~~~~~~~~(D)3
   return (ad[0] + ad[1] + ad[2] + ad[3]) % 256 ;
}

void entry( char* sad , char* hn ) {
    int ad[4] , idx ;

   str_ip4( ad , sad ) ;
   idx = hash( ad ) ;
   table[ idx ] = ihl_cons( ad , hn , table[ idx ] ) ;
}

void main() {
   entry( "192.168.11.2" ,  "perrine" ) ;
   entry( "192.156.145.1" , "www.fukui-nct.ac.jp" ) ;
   printf( "%s\n" , dig_x( "192.168.11.2" ) ) ;
}
```