情報構造論 (斉藤)　後期中間試験　　番号：

4EI　2014/11/27　(1/3)　　氏名：

6 つの選択問題より、5 個を選んで回答せよ。

1　処理トレース (20)

```
int stack[ 10 ] ;
int *sp = stack ;

void push( int x ) {
   *sp++ = x ;
}

int pop() {
   return *--sp ;
}

void main() {
   printf( "%d\n" ,
           eval( "12+4-" ) ) ;
}
```

```
int eval( char* s ) {
   int x , y ;
   for( ; *s != '\0' ; s++ )
      if ( *s >= '0' && *s <= '9' ) {
         push( *s - '0' ) ;
      } else {
         switch( *s ) {
         case '+' : x = pop() ; y = pop() ;
                    push( x + y ) ; break ;
         case '-' : x = pop() ; y = pop() ;
                    push( x - y ) ; break ;
         }
      }            // データ構造の内容変化
   return pop() ;  // を適切に示してあれば
}                  // 中間点あり。
```

1. 上記プログラムの実行結果を答えよ。(15)
2. このプログラムで使われている、データの 出入り順序に由来 する、データ構造 の名称 (略称と順序を表す単語列) を答えよ。(5)

2　ハッシュ法記述問題 (20)

```
#define HSIZE 10
struct NamePhone {
   char name[ 20 ] ;
   int  phone ;
} hash[ HSIZE ] ;
// 登録できる残り件数
int hcount = HSIZE ;
void entry( char n[] , int p ) {
   int idx = hash_func( n ) ;
   if ( hcount > 0 ) {
      while( hash[ idx ].name[ 0 ] != '\0' )
         idx = (idx + 1) % HSIZE ;
      hcount-- ;
      strcpy( hash[ idx ].name , n ) ;
      hash[ idx ].phone = p ;
   }
}
```

```
// ハッシュ関数
int hash_func( char key[] ) {
   int s = 0 , i ;
   for( i = 0 ; key[ i ] != '\0' ; i++ )
      s = s + key[ i ] ;
   return s % HSIZE ;
}
```

```
void main() {
   entry( "t-saitoh" , 111111 ) ;
   entry( "tomoko" ,   112233 ) ;
   printf( "Q: %d\n" ,
           search( "t-saitoh" ) ) ;
   printf( "Q: %d\n" ,
           search( "cat" ) ) ;
}
// 文字列比較: strcmp(a,b)==0
```

```
int search( char n[] ) { // 検索関数 search() を作成せよ。
   // 見つけたら配列番号、見つからない場合は-1 を返す。
   // hcount を適切に処理し、データ 10 件登録時でも正しく動くこと。
```

3　説明問題 (20)

以下の 2 つの説明問題より、一つを選んで回答せよ。
ただし、図なども交え具体的に説明すること。(説明=8, 図=6, 括弧内=6)

1. AVL 木について説明せよ。(検索回数なども説明すること)
2. B 木について説明せよ。(位数と枝に対する条件も説明すること)

4　文字列の 2 分木処理 (20)

```
struct StrTree {                | void strprint( struct StrTree* p ) {
   char* str ;                  |    if ( p != NULL ) {
   struct StrTree* left ;       |       strprint( p->left ) ;
   struct StrTree* right ;      |       printf( "%s " , p->str ) ;
} ;                             |       strprint( p->left ) ;
                                |    }
struct StrTree* top = NULL ;    | }

struct StrTree* stcons( char* s ,    // 文字列 2 分木のコンストラクタ
                        struct StrTree* l ,
                        struct StrTree* r ) {
   struct StrTree* ans ;

   ans =  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(A)3
   if (                        ) {
     ~~~~~~~~~~~~~~~~~~~~~~~~(B)2    | [設問]
      ans->str = s ;                 | 文字列の 2 分岐のプログラムの下線部
      ans->left = l ;                | について、空欄は適切な処理を埋め、
      ans->right = r ;               | (D),(E) は型を答えよ。
   }                                 |
   return ans ;                      | (D)_____________________________3
}                                    |
                                     | (E)_____________________________3
void strentry( char* s ) {
   struct StrTree** ppt = &top ;             | void main() {
                                             |   strentry( "mitsuki" ) ;
   while(                    ) {             |   strentry( "t-saitoh" ) ;
       ~~~~~~~~~~~~~~~~~~~~(C)3              |   strentry( "ayuka" ) ;
      if ( strcmp( (*ppt)->str , s ) == 0 )  |   strentry( "tomoko" ) ;
         break ;     ~~~~~~~~~~(D)           |   strprint( top ) ;
      else if ( strcmp( (*ppt)->str , s ) > 0 ) | }
         ppt = &( (*ppt)->left ) ;
      else     ~~~~~~~~~~~~~~~~~(E)
         ppt =
   }           ~~~~~~~~~~~~~~~~~~~~~~~~~~(F)3

}  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(G)3
```

5　意志決定木 (20)

2 分木を応用した意志決定木は、質問と Yes の場合の木/No の場合の木と、末端には最終決断が記載されている。設問 4 の 2 分木で Yes は左の枝、No は右の枝とした場合、質問をして最終決断を表示する関数 question() を作成せよ。(5x4)

```
struct StrTree* q
   = stcons( "プログラム好き?" ,
        stcons( "勉強好き?" ,
                stcons( "進学だ" , NULL , NULL ) ,
                stcons( "ソフト系に就職だ" , NULL , NULL ) ) ,
        stcons( "仕事好き?" ,
                stcons( "ハード系に就職だ" , NULL , NULL ) ,
                stcons( "自宅警備員だ" , NULL , NULL ) ) ) ;

void queston( struct StrTree* p ) {
   char buff[ 10 ] ;

   while(                                                ) {
      printf( "%s\n" , p->str ) ;
      scanf( "%s" , buff ) ;
      if ( strcmp( buff , "Yes" ) == 0 ) {


      } else if ( strcmp( buff , "No" ) == 0 ) {


      } else {
         printf( "" ) ;
      }
   }


}
void main() {
   question( q ) ;
}
```

6　2 分木の廃棄 (20)

設問 4 の文字列の 2 分木の処理では、生成した 2 分木のヒープメモリを返却していない。2 分木のメモリを全て返却する関数 stfree(struct StrTree* p) を作成せよ。

使い方は、設問 4 であれば main() の最後に、stfree(top); を呼び出す。

```
void stfree( struct StrTree* p ) { // 以下を作成せよ。
```