情報構造論 (斉藤)　前期期末試験　　番号：　解答例 #1/3

4EI　2018/08/1　(1/3)　　氏名：

以下の 6 つ設問から 5 つを選んで回答せよ。

# 1　再帰動作トレース (20)

```
int find( struct List* p , int key ) {
   for( ; p != NULL ; p = p->next ) {
      printf( "%d-%d\n" , p->data , key ) ;
      ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(A)
      if ( p->data == key )
         return 1 ;
   }
   return 0 ;
}

int set_uniq( struct List* p ) {
   if ( p == NULL )
      return 1 ;
   else if ( find( p->next , p->data ) )
      return 0 ;
   else
      return set_uniq( p->next ) ;
}
void main() {
   struct List* top = cons( 1, cons( 2, cons( 3, cons( 3, NULL ) ) ) ) ;
   if ( !set_uniq( top ) )
      printf( "uniq\n" ) ;
}
```

not-uniq

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

| struct List* cons( int x, struct List* n ) {
|    struct List* ans =
|       (struct List*)malloc( sizeof(struct List) ) ;
|    if ( ans != NULL ) {
|       ans->data = x ;
|       ans->next = n ;
|    }
|    return ans ;
| }
|-[設問]----------------------------
|   このリスト処理のプログラムで、
|   表示される内容を答えよ。(3x6+2)
|-----------------------------------
```

2-1
3-1
3-1
3-2
3-2
3-3
not-uniq

# 2　説明問題 (20)

1. 2 進数を用いた集合計算をするための方法を説明せよ。集合の要素は 0〜30 までの整数とし、$A$={1,2,3},$B$={3,5,7,9}で、$A \cap B, A \cup B$ をどのように計算するか、具体的な値を交えて説明せよ。(12)

2. 上記の計算方法で、差集合 $A - B$ ={1,2} の計算方法を説明せよ。(8)

（1）変数を2進数とした際の各桁のbitを、
集合に含まれる=1,含まれない=0として、
30,…6, 5, 4, 3, 2, 1, 0 の各bitに対応付ける。
この場合、
A = 00,0000,0110 )2 = { 1,2,3 }
B = 10,1010,1000 )2 = { 3,5,7,9 }
集合積 A∩Bは、A & B
集合和 A∪Bは、A | B で表せる。
# A&B, A|B のbit列が示してあればさらに良い。

(2) 集合差は、Bを含んではいけないので
A ∩ ¬B (A and (not B))
C言語の2進数演算であれば、

A & ~B

で計算すればいい。

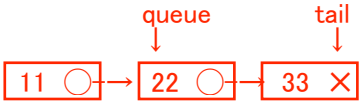
## 3　データイメージ図 (20)

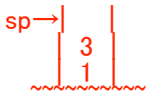
以下のようなプログラムを作成した。設問に答えよ。

```
int stack[ 100 ] ;          | struct List* queue = NULL ;
int* sp = stack ;           | struct List** tail = &queue ;
                            |                      ~~~~~~(A)
void push( int x ) {        | void put( int x ) {
   * sp = x ;               |    *tail = cons( x , NULL ) ;
     ~~(B)                  |    tail = &( (*tail)->next ) ;
   sp++ ;                   | }          ~~~~~~~~~~~~~~~~~~(C)
}                           |
int pop() {                 | int get() {
   sp-- ;                   |    struct List* d = queue ;
   return *sp ;             |    int ans = d->data ;
}         ~~~(D)            |    queue = queue->next ;
                            |            ~~~~~~~~~~~(E)
                            |    free( d ) ;
void main() {               |    return ans ;
   push( 1 ) ;              | }                    [2]
   push( 2 ) ;
   printf( "%d\n" , pop() ) ; /* (F) */      (A)____List **_____________
   push( 3 ) ;
   put( 11 ) ;                               (B)____int *______________
   put( 22 ) ;
   printf( "%d\n" , get() ) ; /* (G) */      (C)____List **_____________
   put( 33 ) ;
}  ~~~~~~~~~(H)                              (D)_____int_______________
   [1]     (F)___2_________  (G)___11_____   (E)____List *_____________
```

間違った解答では、リストによるstackの図で説明していたり、リングバッファのqueueの図を書いていたり、プログラムのデータ構造を見ていない解答が目立った。

1. 行 (F),(G) の出力結果を答えよ。(2x2)
2. 下線部 (A)〜(E) の型を答えよ。(2x5)
3. 下線 (H) の処理終了後のメモリの状態を図を用いて示せ。
   図の解答は余白部に記入。(2x3)




## 4　説明問題 (20)

1. 「配列は○○だがリストは△△である」といった対比した利点欠点を、２つ答えよ。(10)
2. 循環リストの構造と用途を、図や簡単な事例を交えて説明せよ。(10)

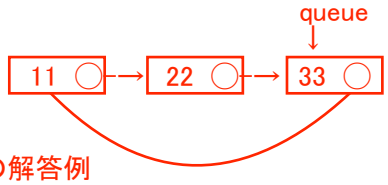
設問1 の解答例

(1) 配列は、有限サイズだが、
リストは、必要に応じて確保するため
無限にサイズを伸ばすことができる。

(2) 配列は、途中へのデータ挿入が、
O(N) かかるが、リストは O(1) で可能。

(3) 配列は、N番目の取り出しが、
O(1) だが、リストは、O(N)

(4) 配列は、プログラムが単純で
わかりやすいが、リスト構造は複雑。




設問2の解答例

待ち行列を実装する場合、先頭ポインタと末尾データのポインタを覚える代わりに末尾の next を NULL とせずに、先頭を指すようにする。

プロセス待ち行列を扱う場合に利用される。

## 5　リスト処理の穴埋め

リスト構造を使った集合差を計算するプログラムを作りたい。以下の下線部を埋めよ。

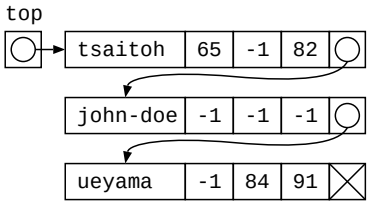
```
struct List*  set_diff( struct List*  a , struct List*  b ) {
~~~~~~~~~~~~~~~~(A)  ~~~~~~~~~~~~~~~~(B) ~~~~~~~~~~~~~~~~(C)
   struct List* ans = NULL
                      ~~~~~~~~~~~~~~~(D)
   for( ; a != NULL ; a = a -> next ) {   # a の全要素を試すループ
      ~~~~~~~~~~~~~~(E)~~~~~~~~~~~~~~(F)
      struct List* p ;

      for( p = b ; p != NULL ; p = p->next ) {
             ~~~~~~~~~~~~(G)
         if ( a->data == p->data )   # a->data が b に含まれるか？
              ~~~~~~~~~~~~~~~~~~(H)
            break ;
         }                          # for( p.. ) が最後まで回った
         if ( p == NULL )           # つまり、b に含まれなかった
              ~~~~~~(I)
            ans = cons( p->data , ans ) ;   # ans の先頭に、p->data を挿入
                  ~~~~~~~~~~~~~~~~~~~~~~(J)
   }
   return ans ;
}
void main() {
   struct List* a = cons( 1 , cons( 2 , cons( 3 , NULL ) ) ) ;
   struct List* b = cons( 3 , cons( 5 , cons( 7 , NULL ) ) ) ;

   print( set_diff( a , b ) ) ; // print() はリスト全要素表示関数
}
```

## 6　課題理解確認 (20)

3 科目のテストの点数 (max100) と名前のデータをリスト構造で保存し、以下のような処理を行いたい。
各データ 1 人毎の最低点を求め、各人の最低点が全員の中で最低の人の名前を求めたい。ただし点数欄には、テストを受けなかった場合には-1 が保存されている。この例では、tsaitoh を返すこと。john-doe は全科目を受けていないので、該当しない。




次のような使い方をして、名前を表示できること。リストのデータを生成する処理は記述しなくてよい。

```
printf( "%s\n" , min_min_user( top ) ) ;
```

```
struct NamePointList {
   char name[ 20 ] ;
   int   point[ 3 ] ;
   struct NamePointList* next ;
} ;
```

JavaScript ユーザへの注意

o char name ; // 1文字だけ？

o 局所変数返しはダメ。
https://www.ei.fukui-nct.ac.jp
/2018/08/06/return-local-array/

```
char* min_min_user( struct NamePointList* p ) {
   struct NamePointList* pm = NULL ; // 最低データ
   int  min_point = 100 ; // 仮りの最低点

   for( ; p != NULL ; p = p->next ) {
      for( int i = 0 ; i < 3 ; i++ )
         if ( p->point[ i ] >= 0
              && p->point[ i ] < min_point ) {
            pm = p ;                    // 最低データを保存
            min_point = p->point[ i ] ; // 最低点の更新
         }
   }
   if ( pm == NULL )
      return NULL ; // 該当者ナシ
   else
      return pm->name ;
}
```