基礎ソフト実験Ｉ　レポート１

締切：10月19日（水）正午

問１.プログラム解説

次は，ある６８０００のアセンブラプログラムの一部分を取り出したものである.
それぞれが，要するに何を行っているか，なるべく分かりやすく，詳しく（２～３行程度）説明せよ.（他人の解答をほぼ丸写ししている場合は，マイナス点を与える）

(1)

```
        MOVE.W (%A0)+,(%A1)+
```

(2)

```
.equ TOP, 0xFFFC00
.equ MASK, TOP+0x80
        MOVE.W  #0x07F7, MASK
```

(3)

```
        .dc.b   'a','b','c','d','e',0
```

(4)

```
.equ BOTTOM,          4
        MOVE.L %A2,BOTTOM(%A0)
```

問２.選択ソート

下記に示したプログラムについて，次の問いに解答せよ.

(1)フローチャートを書け

(2)実際に，実習室でこのプログラムを打ち込んで（もちろんコメント部分を打ち込む必要は無い）動作させ，実行結果の報告を行え．画面のハードコピー，あるいは，実行結果を手で書き写したものを解答すること．確かに正しく動いていることが分かるような簡単な説明があると望ましい.

(3)余裕があれば，感想，意見，改良すべき点を述べよ（内容に応じて得点を与える）

問２のプログラム

・プログラムの目的

与えられたデータを小さい順にソートする．ソートの方法は選択法である。レジスタ

d0,d1 は繰り返しのためのカウンター．レジスタ d2 は対象とするデータ列の中の最小値を保持するレジスタ．レジスタ a3 は最小値のデータの場所（アドレス）を表す。整列するデータ列の先頭アドレスを格納しているレジスタ a1 から順にデータの大小を調べ、データ列の最後までの中で最も小さい値の場所を見つける（内側のループ)。そして、その最小値とレジスタ a1 のアドレスに格納されている値と入れ替える。これを整列対象の先頭から順番に行う（外側のループ）ことによって、小さい順に整列を行うことができる。詳しくはアルゴリズムに関する本を参照せよ．また，参考として，末尾にＣ言語で作成した等価動作を行うプログラムを載せる.

```
**
** Selection Sort
** ex2.s
**

.section .text
** Main Routine
start:
        lea.l   DATA,%a1        /* a1はソートするデータの先頭を指す */
        jsr     SELECTION       /* 選択ソートのサブルーチンに分岐 */
        stop    #0x2700         /* 終了 */

** Sub Routine
SELECTION:
        movem.l %d0-%d2/%a1-%a3,-(%a7)    /* レジスタの退避 */
        moveq.l #LENGTH,%d0
        subq.w  #1,%d0          /* d0は外側のループの繰返回数 = LENGTH - 1 */

LOOP2:
        move.w  %a1,%a3         /* a3は最小値のデータの場所を表す */
        move.w  (%a3),%d2       /* d2は最小値のデータを表す */
        move.w  %a1,%a2
        adda.w  #2,%a2
        move.w  %d0,%d1         /* d1 = d0 */

LOOP1:
        cmp.w   (%a2),%d2
```

```
        bcs     LABEL1
        move.w  %a2,%a3
        move.w  (%a3),%d2

LABEL1:
        adda.w  #2,%a2          /* a2 <- a2 + 2 */
        subq.w  #1,%d1          /* d1 <- d1 - 1 */
        bne     LOOP1           /* 内側のループの終了判定 */

        move.w  (%a1),(%a3)     /* swap(a1,a3) */
        move.w  %d2,(%a1)

        adda.w  #2,%a1
        subq.w  #1,%d0
        bne     LOOP2           /* 外側のループの終了判定 */

        movem.l (%a7)+,%d0-%d2/%a1-%a3   /* レジスタの回復 */
        rts

** Data Area
.section .data
        .equ    LENGTH, 7
DATA:   .dc.w   9,5,3,7,6,4,8

.end

/*************************************/
/*          C言語版　選択ソート
/*************************************/

#include <stdio.h>

#define LENGTH 7

void selection( int* a1 )
{
```

```
    int *a2, *a3;
    int d0, d1, d2;

    d0 = LENGTH - 1; /* d0 は外側のループ回数 ＝ LENGTH - 1 */

    /* LOOP2 */
    while ( d0 > 0 ) {

a3 = a1; /* a3 は最小値のデータの場所を表す */
d2 = *a3; /* d2 は最小値のデータを表す */
a2 = a1 + 1;     /* 「ポインタ変数 + 1」は「次の要素」の意味 */

d1 = d0;
/* LOOP1 */
while ( d1 > 0 ) {
    fprintf( stderr, "a2 = 0x%p, *a2 = %d¥n", a2, *a2 );
    if ( ! ( (*a2) > d2 ) ) {
a3 = a2;
d2 = *a3;
    }
    /* LABEL1 */
    a2 = a2 + 1; /* 「ポインタ変数 + 1」は「次の要素」の意味 */
    d1--;
}

/* swap(a1,a3) */
*a3 = *a1;
*a1 = d2;

a1 = a1 + 1;      /* 「ポインタ変数 + 1」は「次の要素」の意味 */
d0--;
    }
}

main()
{
```

```
    int DATA[] = { 9, 5, 3, 7, 6, 4, 8 };
    int* a1;

    /* start */
    a1 = DATA; /* a1 はソートするデータの先頭を指すポインタ変数 */
    selection( a1 );

    exit(0);
}
```

問３.構造体(住所録)へのアクセス

下記に示したプログラムについて，次の問いに解答せよ.

(1)このプログラムでの，メモリ内のデータの配置を図示せよ
(2)実際に，実習室でこのプログラムを打ち込んで（もちろんコメント部分を打ち込む必要は無い）動作させ，実行結果の報告を行え．画面のハードコピー，あるいは，実行結果を手で書き写したものを解答すること．確かに正しく動いていることが分かるような簡単な説明があると望ましい.
(3)余裕があれば，感想，意見，改良すべき点を述べよ（内容に応じて得点を与える）

問３のプログラム

・プログラムの目的

NAME(20byte),ADDRESS(40byte)からなる構造体の要素を取り出すサブルーチンとサンプルプログラム

・プログラムのポイント

構造体の要素にはディプレースメント付きアドレスレジスタ間接形式を用いてアクセスするのが簡単な実現方法ではあるが、本プログラムでは文字列を書き出すため、ポストインクリメント付きアドレスレジスタ間接形式を用いて各文字にアクセスするようにした

```
************************************************
.section .text
start:
```

```
        move.l #1,%d0     /* STRUCT の引数 */
        jsr    STRUCT     /* STRUCT(%d0.l) → NAME_DATA,ADDRESS_DATA */
        stop   #0x2700    /* プログラム終了 */

************************************************
** 構造体の構造
************************************************
.equ        NAME,0       /* NAME は先頭から 0byte 目 */
.equ        ADDRESS,20   /* ADDRESS は先頭から 20byte 目 */
.equ        SIZE,60      /* 一人分のデータサイズは 60byte */

************************************************
** 構造体の要素を取り出すサブルーチン
**
** 引数     %d0.l = 構造体のインデックス番号
** 戻り値   NAME_DATA(名前のデータ)
**          ADDRESS_DATA(住所のデータ)
************************************************
STRUCT:
        * レジスタ退避
        movem.l  %d0-%d7/%a0-%a6,-(%sp)

        mulu    #SIZE,%d0           /* d0.l=取り出すインデックス番号×60 */
        add.l   #ADDRESS_BOOK,%d0   /* d0.l=取り出す要素の先頭アドレス */
        movea.l %d0,%a0             /* a0.l=d0.l */

        moveq.l #0,%d1              /* d1.l=0 ループカウンタの初期化*/
        lea.l   NAME_DATA,%a2       /* 書き出す領域の先頭アドレス */
        movea.l %a0,%a1             /* a1.l=a0 */
NAME_LOOP:      /* NAME を書き出すルーチン */
        move.b  (%a1)+,(%a2)+       /* NAME の書き出し */
        addq.l  #1,%d1              /* ループカウンタ++ */
        cmp.l   #20,%d1             /* ループカウンタ<20 ならば */
        bne     NAME_LOOP           /* NAME_LOOP へ */

        moveq.l #0,%d1              /* d1.l=0 ループカウンタの初期化 */
```

```
        lea.l   ADDRESS_DATA,%a2     /* 書き出す領域の先頭アドレス */
        movea.l %a0,%a1              /* a1.l=a0.l */
        adda.l  #ADDRESS,%a1         /* a1.l=a1.l+ADDRESS */
ADDRESS_LOOP:   /* ADDRESS を書き出すルーチン */
        move.b  (%a1)+,(%a2)+        /* ADDRESS の書き出し */
        addq.l  #1,%d1               /* ループカウンタ++ */
        cmp.l   #40,%d1              /* ループカウンタ<40 ならば */
        bne     ADDRESS_LOOP         /* ADDRESS_LOOP へ */

        * レジスタの復帰
        movem.l (%sp)+,%d0-%d7/%a0-%a6
        rts                          /* サブルーチンから復帰*/

*************************************************
** 住所録
** NAME の長さ 20byte
** ADDRESS の長さは 40byte
*************************************************
.section .data
ADDRESS_BOOK:
        .ascii  "TAROU               "    /* NAME */
        .ascii  "HUKUOKASHI HIGASHIKU                    "  /* ADDRESS */

        .ascii  "HANAKO              "    /* NAME */
        .ascii  "HUKUOKASHI MINAMIKU                     "  /* ADDRESS */

*************************************************
** STRUCT の出力
*************************************************
NAME_DATA:      .ds.b 20                /* NAME 出力先 */
ADDRESS_DATA:   .ds.b 40                /* ADDRESS 出力先*/

.end
```