プログラミング Ⅰ

Report#7

提 出 日:2006 年某日(木)

所 属 :工学部情報工学科

学籍番号: 065702G

氏名 : 新 垣 智 規

◎問題. 第 2 回試験問題より、解答確認のプログラムを作成し考察せよ。
併せて、構造体。共有体についても考察せよ。

◎ソースコード(test1.c)

```
#include <stdio.h>


#define N1   7

#define N2   3

#define S1   *

#define S2   /

#define MAX  24


main()

{

  int    iq[50], i;

  float  fq[50];

  char work[MAX], *dest, *src;

  static char cbuf[] = "var test program";

  static int  ibuf[] = {1,2,3,5,8,13};


  iq[ 1] = N1;

  iq[ 2] = N2;

  iq[ 3] = N1 S2 N2;

  iq[ 4] = N1 % N2;

  iq[ 5] = (N1+N2) S1 (N1-N2);

  fq[ 6] = (float)N1 / (float)N2;

  fq[ 7] = N1 / N2;

  iq[ 8] = N1 - N1 % N2;

  iq[ 9] = N1 > N2;

  iq[10] = N1 < N2;

  iq[11] = (N1 > N2) || !(N1 < N2);

  iq[12] = !(N1 > N2) || (N1 < N2);
```

```
  iq[13] = N1 | N2;
  iq[14] = N1 & N2;
  iq[15] = N2 << 2;
  iq[16] = N1*N1 >> 1 + N2;
  iq[17] = sizeof(work);
  for(iq[18] =0, src=cbuf, dest=work; *src != NULL; iq[18]++,src++, dest++){
    *dest = *src;
  }

  iq[19] = src - cbuf;
  iq[20] = cbuf[N1] - cbuf[N2-1];
  iq[21] = cbuf[N1] - cbuf[0];
  iq[22] = cbuf[9] - cbuf[5];
  iq[23] = ibuf[1] * ibuf[4];
  iq[24] = ibuf[5] - ibuf[3];
  src = cbuf;
  iq[25] = *(src+4) - 'b' ;

  for(i=1; i<=5; i++) printf("q[%2d]=%4d,%c",i,iq[i],(i%5!=0? ' ':'\n'));
  for(i=6; i<=7; i++) printf("q[%2d]=%4.1f,%c",i,fq[i],(i%5!=0? ' ':'\n'));
  for(i=8; i<=25; i++) printf("q[%2d]=%4d,%c",i,iq[i],(i%5!=0? ' ':'\n'));

  return(0);
}
```

◎ 実行結果

```
q[ 1]=   7, q[ 2]=   3, q[ 3]=   2, q[ 4]=   1, q[ 5]=  40,
q[ 6]= 2.3, q[ 7]= 2.0, q[ 8]=   6, q[ 9]=   1, q[10]=   0,
q[11]=   1, q[12]=   0, q[13]=   7, q[14]=   3, q[15]=  12,
q[16]=   3, q[17]=  24, q[18]=  16, q[19]=  16, q[20]=   2,
q[21]=  -2, q[22]=  11, q[23]=  16, q[24]=   8, q[25]=  18,
```

◎ソースコード(test2.c)

```
#include <stdio.h>

int main(){

  /* Q1 : &と*、ポインタ演算子 */
  {
    int a =2, b = 3, c = 5, *p,*q;

    p = &b; q = &c; a = *p + *q;

    printf("Q01 = %d\n", a);
  }

  /* Q2 : アドレスと値の代入 */
  {
    int a = 2, *p;

    p = &a; *p = 5;

    printf("Q02 = %d\n",a);
  }

  /* Q3 : 配列とポインタ */
  {
    int m[5] = {1,5,2,4,3}, *p;

    p = m;

    printf("Q03 = %d\n", *m);
    printf("Q04 = %d\n", *(m+1));
    printf("Q05 = %d\n",p[0]);
```

```
  printf("Q06 = %d\n", p[2]);
}

/* Q04 : int 型 1 次配列 */
{
  int m[5] = {10,20,40,50,30};

  printf("Q07 = %d\n",*m);
  printf("Q08 = %d\n",*(m+3));
  printf("Q09 = %d\n",*m+3);
  printf("Q10 = %d\n",*m+*(m+3));
}

/* Q05 : int 型 2 次元配列 */
{
  int d[4][3] = {{1,2,3},{5,6,7},{4,6,8},{9,7,5}};

  printf("Q11 = %d\n",*d[2]);
  printf("Q12 = %d\n",*(d[2]+2));
  printf("Q13 = %d\n",*d[2]+2);
  printf("Q14 = %d\n",**d);
  printf("Q15 = %d\n",*(*d+3));
  printf("Q16 = %d\n",**d+6);
  printf("Q17 = %d\n",*(d[1]+2));
  printf("Q18 = %d\n",**(d+2));

}

/* Q06 : ポインタと文字列 */
{
  char *str = "abcdefg", *p;

  p = str +3;
```

```
  printf("Q19 = %s\n",p);
}
/* Q7：配列、ポインタと文字列 */
{
  char m[] = "abcdefghi";
  char *p, *q;

  p = "jklmnopq";

  printf ("Q20 = %c\n", *p);
  printf ("Q21 = %c\n", *(p+2));

  p = &m[0]; q = m;

  printf ("Q22 = %c\n", *m);
  printf ("Q23 = %c\n", *(p+1));
  printf ("Q24 = %c\n", *q+2);

  p = &m[3];

  printf ("Q25 = %c\n", *p);
  printf ("Q26 = %c\n", *(m+4));
  printf ("Q27 = %c\n", *m+5);
}

/* Q8：ポインタ、文字変換 */
{
  char *p,m[]="abcde";

  p = m; *(p + 2) = 'x';
  printf ("Q28 = %s\n", p);
```

```
  *(p + 3) = 'z';
  printf ("Q29 = %s\n", p);

  p = m;
  printf ("Q30 = %s\n", p);
}

/* Q9：ポインタと配列と文字列 */
{
  char *q[] = {"abcd", "12345", "ABCDEFG", "987"};

  printf ("Q31 = %c\n", *q[2]);
  printf ("Q32 = %c\n", q[3][2]);
  printf ("Q33 = %c\n", *(q[2]+2));
  printf ("Q34 = %c\n", *(*(q+3)+2));
  printf ("Q35 = %c\n", **(q+1));
}

/* Q10：配列とポインタ */
{
  char m[6] = {'a','b','c','d','e','f'};
  char mm[2][3] = {'a','b','c','d','e','f'};

  printf ("Q36 = %c\n", m[4]);
  printf ("Q37 = %c\n", *(m+2));

  printf ("Q38 = %c\n", mm[0][0]);
  printf ("Q39 = %c\n", mm[1][1]);
  printf ("Q40 = %c\n", mm[0][2]);

  printf ("Q41 = %08x (１次元配列 m[]の先頭アドレス)\n", &m[0]);
  printf ("Q42 = %08x (１次元配列 m[]の先頭アドレス)\n", m);
```

```
    printf ("Q43 = %c （１次元配列 m[]の先頭の値）\n",m[0]);
    printf ("Q44 = %c （１次元配列 m[]の先頭の値）\n",*m);

    printf ("Q45 = %08x （２次配列 mm[][]の先頭アドレス)\n", &mm[0][0]);
    printf ("Q46 = %08x （２次配列 mm[][]の先頭アドレス)\n", mm[0]);
    printf ("Q47 = %08x （２次配列 mm[][]の先頭アドレス)\n", &**mm);
    printf ("Q48 = %08x （２次配列 mm[][]の先頭アドレス)\n", *mm);

    printf ("Q49 = %c （２次配列 mm[][]の先頭の値）\n",mm[0][0]);
    printf ("Q50 = %c （２次配列 mm[][]の先頭の値）\n",*mm[0]);
    printf ("Q51 = %c （２次配列 mm[][]の先頭の値）\n",**mm);

    printf ("Q52 = %c （２次配列 mm[2][3]の４番目の値）\n",mm[1][0]);
    printf ("Q53 = %c （２次配列 mm[2][3]の４番目の値）\n",*mm[1]);
    printf ("Q54 = %c （２次配列 mm[2][3]の４番目の値）\n",*(*mm+3));
    printf ("Q55 = %c （２次配列 mm[2][3]の４番目の値）\n",**(mm+1));
  }

  return(0);
}
```

◎実行結果

```
Q01 = 8
Q02 = 5
Q03 = 1
Q04 = 5
Q05 = 1
Q06 = 2
Q07 = 10
Q08 = 50
Q09 = 13
Q10 = 60
Q11 = 4
```

Q12 = 8

Q13 = 6

Q14 = 1

Q15 = 5

Q16 = 7

Q17 = 7

Q18 = 4

Q19 = defg

Q20 = j

Q21 = l

Q22 = a

Q23 = b

Q24 = c

Q25 = d

Q26 = e

Q27 = f

Q28 = abxde

Q29 = abxze

Q30 = abxze

Q31 = A

Q32 = 7

Q33 = C

Q34 = 7

Q35 = 1

Q36 = e

Q37 = c

Q38 = a

Q39 = e

Q40 = c

Q41 = bffffa84（1 次元配列 m[]の先頭アドレス）

Q42 = bffffa84（1 次元配列 m[]の先頭アドレス）

Q43 = a （1 次元配列 m[]の先頭の値）

Q44 = a （1 次元配列 m[]の先頭の値）

Q45 = bffffa8a（２次配列 mm[][]の先頭アドレス）

Q46 = bffffa8a（２次配列 mm[][]の先頭アドレス）

Q47 = bffffa8a（２次配列 mm[][]の先頭アドレス）

Q48 = bffffa8a（２次配列 mm[][]の先頭アドレス）

Q49 = a （２次配列 mm[][]の先頭の値）

Q50 = a （２次配列 mm[][]の先頭の値）

Q51 = a （２次配列 mm[][]の先頭の値）

Q52 = d （２次配列 mm[2][3]の４番目の値）

Q53 = d （２次配列 mm[2][3]の４番目の値）

Q54 = d （２次配列 mm[2][3]の４番目の値）

Q55 = d （２次配列 mm[2][3]の４番目の値）

◎考察

今回のテストのプログラムの、解答を確認するためのプログラム。

ほとんどは指示されているので、すべてをタイプし、実行した。

test2.c の Q41~Q55 までは実際の解答を組み込み、アドレスは%06x を使って 16 進数、値は%c を使い 1 文字表示した。

アドレスについては、図解する。

構造体についての考察

◎ソースコード(struct.c)

```
/*                                                                    Program    : struct.c
Comment    : 構造体による閏年判断                                          */

#include <stdio.h>

struct smp{
  int   day;
  int   month;
  int   year;
}date,*pdate;

isleap1(struct smp d){
  int r4,r100,r400;

  r4   = d.year % 4;
  r100 = d.year % 100;
  r400 = d.year % 400;

  return ( ((r4 == 0) && (r100 != 0)) || (r400 == 0) );
}

isleap2(struct smp *d){
  int r4,r100,r400;

  r4   = d->year % 4;
  r100 = d->year % 100;
  r400 = d->year % 400;

  return ( ((r4 == 0) && (r100 != 0)) || (r400 == 0) );
}
```

```
main(){
  date.day   = 24;
  date.month = 2;
  date.year  = 1900;

  printf("%4d 年は閏年で%s\n",date.year,
         (isleap1(date) != 0)? "す。":"ない。");
  printf("%x %x %x\n",&date.day,&date.month,&date.year);

  printf("%4d 年は閏年で%s\n",date.year,
         (isleap2(&date) != 0)? "す。":"ない。");
  printf("%x %x %x\n",&date.day,&date.month,&date.year);
}
```

◎ 実行結果

```
1900 年は閏年でない。
30d0 30d4 30d8
1900 年は閏年でない。
30d0 30d4 30d8
```

◎考察

このプログラムは構造体を理解するためのサンプルプログラムを実行した結果である。

ここでは、閏年の定義を決め、1900 年が閏年であったか否かを調べるプログラムを作成している。

ここでの閏年の定義は 100 の倍数でなく、かつ 4 の倍数及び 400 の倍数であるとしている。

main 関数の前に isleap1 と isleap2 で閏年の判別をし、メインではそれを呼び出している。printf で書き出している”す。”と”ない。”は「:」で区切られている。この:を使って、閏年であるか無いかと区別している。入力された年数が真であるなら（閏年であるなら）、”す。”を、偽なら（閏年でないなら）”ない。”を出

力する。また、改行してアドレスを表示しているが、int 型を使っているので各々 4byte の領域を確保していることが分かる。
また、この isleap2 ではポインタを使っており、「->」を使って date を year に取り込んでいる。

共用体についての考察

◎ソースコード

```
#include <stdio.h>


union smp{
  char  b08;
  short b16;
  int   b32;
};


main(){
  union smp var;


  var.b08=2;
  puts("---------------");
  puts("var.b08=2");
  printf("var.b08=0x      %02x=%d\n",var.b08,var.b08);
  printf("var.b16=0x    %04x=%d\n",var.b16,var.b16);
  printf("var.b32=0x%08x=%d\n",var.b32,var.b32);
  printf("var ADDRESS\n");
  printf("var.b08=0x%x +=%%0x%x\n",&var.b08,&var.b08+1);
  printf("var.b16=0x%x +=%%0x%x\n",&var.b16,&var.b16+1);
  printf("var.b32=0x%x +=%%0x%x\n",&var.b32,&var.b32+1);
```

```
  var.b16=260;
  puts("---------------");
  puts("var.b16=260");
  printf("var.b08=0x      %02x=%d\n",var.b08,var.b08);
  printf("var.b16=0x    %04x=%d\n",var.b16,var.b16);
  printf("var.b32=0x%08x=%d\n",var.b32,var.b32);

  var.b32=66000;
  puts("---------------");
  puts("var.b32=66000");
  printf("var.b08=0x      %02x=%d\n",var.b08,var.b08);
  printf("var.b16=0x    %04x=%d\n",var.b16,var.b16);
  printf("var.b32=0x%08x=%d\n",var.b32,var.b32);
}
```

◎実行結果

```
---------------
var.b08=2
var.b08=0x      02=2
var.b16=0x    02e0=736
var.b32=0x02e073c4=48264132
var ADDRESS
var.b08=0xbffffab8 +=%0xbffffab9
var.b16=0xbffffab8 +=%0xbffffaba
var.b32=0xbffffab8 +=%0xbffffabc
---------------
var.b16=260
var.b08=0x      01=1
var.b16=0x    0104=260
var.b32=0x010473c4=17068996
---------------
var.b32=66000
```

```
var.b08=0x      00=0

var.b16=0x    0001=1

var.b32=0x000101d0=66000
```

◎ 考察

この共有体というものは、アドレスを共有している。

それを証明しているのが上記のサンプルプログラムである。

まず、char、short、int 等の違う型を使い、それぞれを b08、b16、b32 と宣言する。そして、b08、b16、b32 に各々値を代入する。

そのときのアドレスがどうなっているかを表示しているのが、この実行結果だ。

例えば b08=2 を代入した時、16 進数表示で b08 のアドレスは「02」となる。

ここで他の値を見てみると、b16 は「02e0」、b32 は「0x02e073c4」となっている。

ここで注目する点は、アドレスで b08 も b16 も b32 もすべて始まりのアドレスは同じということだ。つまり、b02 のアドレスを b16 と b32 が、b16 のアドレスを b32 が共有している、ということになる。

詳しくは後に図示するのでそれを参照。