1　2016 年 情報科学 1 期末試験

1.1　問題 1

(プログラムコード: fp-toi1.c)

```
1  //解答 (1)2 (2)1 (3)6 (4)6 (5)1 (6)4 (7)4
2
3  #include<stdio.h>
4
5  int main()
6  {
7    int a=20, b=123;
8    double e=10.0,f=3.14;
9    char g='a'; //ASCII コード表を参照 16 進数で 61, 10 進数に変換すると 97
10
11   printf("9l:%3d%3d\n", a, b);
12   printf("10l:%3d\n", a % b);
13   printf("11l:%5.2f%5.2f\n", e,f);//5 桁の小数点弟 2 位
14   printf("12l:%d\n", g-3); // 10 進数 97-3=94
15   printf("13l:%c\n", g-3); // ASCII コード表を参照 文字 a-3=^
16   printf("14l:%x\n", g-3); // 16 進数 61-3=5e %X(大文字) なら 5E
17   printf("15l:%p\n",&a);
18   printf("16l:%p\n",&a+1); // int は 4 バイトのメモリ領域をとるので、15 行目で出力したアドレスから 4 増えたアドレスとなる
19
20   return 0;
21 }
```

1.2 問題 2

(プログラムコード: fp-toi2.c)

```
1  //解答 (1)00 (2)06 (3)4 (4)2 (5)3
2
3  #include<stdio.h>
4
5  int main()
6  {
7    int i;
8
9    for(i = 5; i < 17; i += 2){ //i の値は 5 から始まり 7,9,11,13,15 と 2 ずつ加算される
10     if(i ==8){
11       break; //i の値が 8 になることはないため break はしない
12     }
13     printf("i=%d\n",i);
14   }
15
16   return 0;
17 }
```

1.3 問題 3

(プログラムコード: fp-toi3.c)

```
1  //解答 (1)1 (2)0 (3)1 (4)0 (5)8 (6)4 (7)0 (8)5 (9)3 (10)3 (11)1
2
3  #include<stdio.h>
4
5  int Swap1(int a, int b);
6  int Swap2(int *a, int *b);
7
8  int main()
9  {
10   int a = 3, b = 4, c = 5, d = 9;
11
12   c = Swap1(a, b); // c = 3
13   printf("a=%d, b=%d, c=%d, d=%d\n", a, b, c ,d); // a=3, b=4, c=3. d=9
14
15   d = Swap2(&a, &b); // d=3
16   printf("a=%d, b=%d, c=%d, d=%d\n", a, b, c, d); // a=4, b=3, c=3, d=3
17
18   return 0;
19 }
20
21 // a と b の入れ替えはこの Swap1 関数内でのみ
22 // a の値 3 が代入された tmp を返す
23 int Swap1(int a, int b){
24   int tmp;
25   tmp = a;
26   a = b;
27   b = tmp;
28   return tmp;
29 }
30
31 // a の値 3 が代入された tmp を返す
32 // またアドレスが渡すことで呼び出し元の関数 (main) の変数の値の読み書きが可能
33 // よって main 関数でも a と b の値は入れ替わっている
34 int Swap2(int *a, int *b){
35   int tmp;
36   tmp = *a;
37   *a = *b;
38   *b = tmp;
39   return tmp;
40 }
```

1.4 問題 4

(プログラムコード: fp-toi4.c)

```
1  //解答 (1)0 (2)5 (3)3 (4)6 (5)7 (6)2 (7)2 (8)3 (9)0 (10)3 (11)4
2
3  #include<stdio.h>
4  #include<string.h>
5
6  // 構造体の定義
7  struct PhonesTag{
8    char name[32];
9    char tel[32];
10   int gid;
11 };
12
13 int InputPhones(struct PhonesTag *pPhone);
14 void printPhones(int num, struct PhonesTag *pPhone);
15
16 int main(){
17   struct PhonesTag phones[32];
18   int num, ret;
19
20   // 最大 32 回入力を受け取る
21   for(num = 0; num < 32; num++){
22     // 電話情報を入力
23     // 正しく入力されれば 0 が返ってくる
24     // name で"end"が入力されると-1 が返ってくる
25     ret = InputPhones(&phones[num]);
26
27     // ret が 0 でなければ (-1 が返ってきた時) 入力を終了する
28     if(ret != 0){
29       break;
30     }
31   }
32
33   // 表示
34   printPhones(num, phones);
35
36   return 0;
37 }
38
39 int InputPhones(struct PhonesTag *pPhone){
40
```

```
41    // 名前を入力
42    printf("Input your name:");
43    scanf(" %[^\n]", pPhone->name);
44
45    // 入力された文字列と文字列"end"を比較
46    // 同じ場合 (strcmp(pPhone->name, "end")==0) は-1 を返す
47    if(strcmp(pPhone->name, "end")==0){
48      return -1;
49    }
50
51    // 電話番号を入力
52    printf("Input your phone number:");
53    scanf(" %[^\n]", pPhone->tel);
54
55    // ID を入力
56    printf("Input group id:");
57    scanf(" %d", &(pPhone->gid));
58    return 0;
59  }
60
61  void printPhones(int num, struct PhonesTag *pPhone){
62    int i;
63    for(i = 0; i < num; i++){
64      printf("**** member %d ****\n", i);
65      printf("name : %s\n", pPhone[i].name);
66      printf("phone : %s\n", pPhone[i].tel);
67      printf("group id : %d\n", pPhone[i].gid);
68    }
69    return;
70  }
```

1.5 問題 6

(プログラムコード: fp-toi6.c)

```
1  //解答 (1)0 (2)1 (3)1 (4)1 (5)2 (6)1 (7)3
2
3  #include<stdio.h>
4
5  // 構造体の定義
6  struct Stack {
7    int data[100]; // スタック配列
8    int stackPointer; // 次に push した時に格納される位置
9  };
10
11 void push(struct Stack *st, int n);
12 int pop(struct Stack *st);
13
14 int main()
15 {
16   int i, n;
17   struct Stack st;
18
19   // スタックと stackPointer の初期化
20   st.stackPointer = 0;
21   for(i = 0; i < 100; i ++){
22     st.data[i] = 0;
23   }
24
25   push(&st, 25); // 25, 0, 0 stackPointer = 1
26   push(&st, 32); // 25, 32, 0 stackPointer = 2
27   n = pop(&st); // 25, 32, 0 stackPointer = 1
28   printf("%d\n", n);
29   push(&st, 55); // 25, 55, 0 stackPointer = 2
30
31   n = pop(&st); // 25, 55, 0 stackPointer = 1
32   printf("%d\n", n);
33   n = pop(&st); // 25, 55, 0 stackPointer = 0
34   printf("%d\n", n);
35   n = pop(&st); // 25, 55, 0 stackPointer = 0 なので「Stack is empty.」 が出力される
36   printf("%d\n", n); // stackPointer = 0 なので-1 が返されるので「-1」が出力される
37
38   for(i = 0; i < 10; i++){
39     push(&st, i);
40   }
```

```
41
42    // 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 が格納される
43    // stackPointer = 10
44
45    push(&st, 10); // 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 stackPointer = 11
46
47    return 0;
48  }
49
50  //   push する対象となるスタックと push する値を受け取る
51  // スタックが一杯の場合は何もしない
52  void push(struct Stack *st, int n){
53
54    if(st->stackPointer == 100){
55      printf("Stack is Full.\n");
56      return; // ここで終了するために return
57    }
58
59    st->data[st->stackPointer] = n;
60    st->stackPointer++;
61  }
62
63
64  // pop する対象となるスタックを受け取り値を返す
65  // スタックがからの場合はメッセージを表示した上で-1 を返す
66  int pop(struct Stack *st){
67    if(st->stackPointer == 0){
68      printf("Stack is empty\n");
69      return -1;
70    }
71
72    st->stackPointer--;
73    return st->data[st->stackPointer];
74  }
```