プログラミング I

Report #2(再)

提出日　：2013年8月1日
所　属　：工学部情報工学科
学籍番号：e135732J
氏　名　：前城　健太郎

scanf()関数による標準入力と基本演算子

1 1234 円の買い物をして 1 万円札を出したときの、お釣りの札と硬貨の枚数を求めるプログラムを作成せよ。

1.2 scanf()関数を用いて、価格と支払い金額を入力せよ。

1.2.1 ソースコードの一部

```
9   #include <stdio.h>
10
11  int main()
12  {
13    int price = 1234, pay = 10000;
14    int balance, amount;
15
16    /******************************** scanf */
17    printf("Price?   => "); scanf("%d",&price);
18    printf("Payment? => "); scanf("%d",&pay);
19    printf("----\n");
20
21    /******************************** balance */
22    balance = pay - price;
23    printf("price = %d, ",price);
24    printf("payment = %d, balance = %d\n",pay,balance);
25    printf("----\n");
26
27    /******************************** 5000-yen */
28    amount  = balance / 5000;
29    balance = balance % 5000;
30    printf("5000-yen note = %d\n",amount);
31
32    /******************************** 2000-yen */
33    amount  = balance / 2000;
34    balance = balance % 2000;
35    printf("2000-yen note = %d\n",amount);
36
37    /******************************** 1000-yen */
38    amount  = balance / 1000;
39    balance = balance % 1000;
40    printf("1000-yen note = %d\n",amount);
41
42    /******************************** 500-yen */
43    amount  = balance / 500;
44    balance = balance % 500;
45    printf("500 -yen coin = %d\n",amount);
46
47    /******************************** 100-yen */
48    amount  = balance / 100;
49    balance = balance % 100;
50    printf("100 -yen coin = %d\n",amount);
51
52    /******************************** 50-yen */
53    amount  = balance / 50;
54    balance = balance % 50;
55    printf("50  -yen coin = %d\n",amount);
56
57    /******************************** 10-yen */
58    amount  = balance / 10;
59    balance = balance % 10;
60    printf("10  -yen coin = %d\n",amount);
61
62    /******************************** 5-yen */
63    amount  = balance / 5;
64    balance = balance % 5;
65    printf("5   -yen coin = %d\n",amount);
66
```

```
67    /****************************** 1-yen */
68    printf("1   -yen coin = %d\n",balance);
69
70    return(0);
71  }
```

1.2.2 出力結果　※変数 "Price","Payment"にはそれぞれ 1234,10000 を入力した.

```
Price?   => 1234
Payment? => 10000
----
price = 1234, payment = 10000, balance = 8766
----
5000-yen note = 1
2000-yen note = 1
1000-yen note = 1
500 -yen coin = 1
100 -yen coin = 2
50  -yen coin = 1
10  -yen coin = 1
5   -yen coin = 1
1   -yen coin = 1
```

1.2.3 考察

a) scanf()関数を用いてお釣りとその枚数を計算するプログラムを作成した.
b) 21-25 行のお釣りの計算において、変数 pay は価格、price には支払った値段、balance にはお釣りが格納される.
c) 27 行目以降のお釣りの枚数の計算では、変数 amount にはそれぞれのお札,コインの枚数、balance にはその枚数分の値段を引いた数が格納される.
d) 変数 price、pay はそれぞれ "214748364"（32 ビットの上限値）までの整数なら計算できるが、それ以上の整数だと正常に計算されなかった.
e) 67 行目からの 1 円玉の計算において、1 で割る必要はないので変数 balance に格納されている数をそのまま表示させた.
f) 変数 price が変数 pay より大きい数字な場合、変数 balance はマイナス値が代入され、それぞれのお釣り、お札と硬貨の枚数もマイナスとして計算された.

1.3 例題の変数名を変え、自分自身で考えた変数名にせよ。

1.3.1 半角数字を含む変数

1.3.1.1 ソースコードの一部

```
9   #include <stdio.h>
10
11  int main()
12  {
13    int nedan1 = 1234, 2nedan = 10000;
14    int 0987, ne1dan;
15
16    /****************************** scanf */
17    printf("Price?   => "); scanf("%d",&nedan1);
18    printf("Payment? => "); scanf("%d",&2nedan);
19    printf("----\n");
20
21    /****************************** balance */
22    0987 = 2nedan - nedan1;
23    printf("price = %d, ",nedan1);
24    printf("payment = %d, balance = %d\n",2nedan,0987);
25    printf("----\n");
26
27    /****************************** 5000-yen */
28    ne1dan  = 0987 / 5000;
```

```
29    0987 = 0987 % 5000;
30    printf("5000-yen note = %d\n",ne1dan);
31
32    return(0);
33  }
```

1.3.1.2 コンパイル時のエラー

```
report@2b.c:13:22: error: expected identifier or '('
  int nedan1 = 1234, 2nedan = 10000;
                     ^
report@2b.c:13:21: error: expected ';' at end of declaration
  int nedan1 = 1234, 2nedan = 10000;
                    ^
                    ;
report@2b.c:14:7: error: expected identifier or '('
  int 0987, ba1lance;
      ^
report@2b.c:18:40: error: invalid suffix 'nedan' on integer constant
  printf("Payment? => "); scanf("%d",&2nedan);
                                       ^
report@2b.c:22:4: error: invalid digit '9' in octal constant
  0987 = 2nedan - nedan1;
   ^
report@2b.c:22:11: error: invalid suffix 'nedan' on integer constant
  0987 = 2nedan - nedan1;
          ^
report@2b.c:24:42: error: invalid suffix 'nedan' on integer constant
  printf("payment = %d, balance = %d\n",2nedan,0987);
                                         ^
report@2b.c:28:3: error: use of undeclared identifier 'ba1lance'
  ba1lance  = 0987 / 5000;
  ^
report@2b.c:28:16: error: invalid digit '9' in octal constant
  ba1lance  = 0987 / 5000;
               ^
report@2b.c:29:4: error: invalid digit '9' in octal constant
  0987 = 0987 % 5000;
   ^
report@2b.c:29:11: error: invalid digit '9' in octal constant
  0987 = 0987 % 5000;
          ^
report@2b.c:30:33: error: use of undeclared identifier 'ba1lance'
  printf("5000-yen note = %d\n",ba1lance);
                               ^
```

1.3.1.3 エラーの意味

13 行目:22 文字目:識別子もしくは'('が無い
13 行目:12 文字目:';'が記述の終わりに無い
14 行目:7 文字目:識別子もしくは'('が無い
18 行目:40 文字目:整数定数に無効な接尾辞'nedan'
22 行目:4 文字目:8 進数では'9'は無効な数字です
22 行目:11 文字目:整数定数に無効な接尾辞'nedan'
24 行目:42 文字目:整数定数に無効な接尾辞'nedan'
28 行目:3 文字目:'ba1lance'は宣言されてない識別子です
28 行目:16 文字目:8 進数では'9'は無効な数字です
29 行目:4 文字目:8 進数では'9'は無効な数字です
29 行目:11 文字目:8 進数では'9'は無効な数字です
30 行目:33 文字目:'ba1lance'は宣言されてない識別子です

1.3.1.4 考察

a) 先頭が数字の変数名だけがエラーとして出力された.
b) 先頭が数字となっている変数名を修正する.

1.3.1.5 修正したソースコードの一部

```
9   #include <stdio.h>
10
11  int main()
12  {
13    int nedan1 = 1234, 2nedan = 10000;
14    int 0987, ne1dan;
15
16    /****************************** scanf */
17    printf("Price?   => "); scanf("%d",&nedan1);
18    printf("Payment? => "); scanf("%d",&2nedan);
19    printf("----\n");
20
21    /****************************** balance */
22    0987 = 2nedan - nedan1;
23    printf("price = %d, ",nedan1);
24    printf("payment = %d, balance = %d\n",2nedan,0987);
25    printf("----\n");
26
27    /****************************** 5000-yen */
28    ne1dan  = 0987 / 5000;
29    0987 = 0987 % 5000;
30    printf("5000-yen note = %d\n",ne1dan);
31
32    return(0);
33  }
```

1.3.1.6 出力結果　※変数"nedan"、nedan2 にはそれぞれ 1234、10000 を入力した

```
Price?   => 1234
Payment? => 10000
----
price = 1234, payment = 10000, balance = 8766
----
5000-yen note = 1
```

1.3.1.7 考察

a) 適切な変数にすることで、狙い通りの出力ができた.

1.3.2 全角日本語を含む変数

1.3.2.1 ソースコードの一部

```
9   #include <stdio.h>
10
11  int main()
12  {
13    int 価格 = 1234, 支払い = 10000;
14    int 差額, 余り;
15
16    /****************************** scanf */
17    printf("Price?   => "); scanf("%d",&価格);
18    printf("Payment? => "); scanf("%d",&支払い);
19    printf("----\n");
20
21    /****************************** balance */
22    差額 = 支払い - 価格;
23    printf("price = %d, ",価格);
```

```
24    printf("payment = %d, balance = %d\n",支払い,差額);
25    printf("----\n");
26
27    /****************************** 5000-yen */
28    余り  = 差額 / 5000;
29    差額 = 差額 % 5000;
30    printf("5000-yen note = %d\n",余り);
31
32    return(0);
33  }
```

1.3.2.2 コンパイル時のエラー

```
report@2b2.c:13:7: error: expected identifier or '('
  int 価格 = 1234, 支払い = 10000;
      ^
report@2b2.c:14:7: error: expected identifier or '('
  int 差額, 余り;
      ^
report@2b2.c:17:39: error: expected expression
  printf("Price?   => "); scanf("%d",&価格);
                                      ^
report@2b2.c:18:39: error: expected expression
  printf("Payment? => "); scanf("%d",&支払い);
                                      ^
report@2b2.c:22:3: error: expected expression
  差額 = 支払い - 価格;
  ^
report@2b2.c:23:25: error: expected expression
  printf("price = %d, ",価格);
                        ^
report@2b2.c:24:41: error: expected expression
  printf("payment = %d, balance = %d\n",支払い,差額);
                                        ^
report@2b2.c:28:3: error: expected expression
  余り  = 差額 / 5000;
  ^
report@2b2.c:29:3: error: expected expression
  差額 = 差額 % 5000;
  ^
report@2b2.c:30:33: error: expected expression
  printf("5000-yen note = %d\n",余り);
                               ^
```

1.3.2.3 エラーの意味

13 行目:7 文字目:識別子もしくは'('が無い
14 行目:7 文字目:識別子もしくは'('が無い
17 行目:39 文字目:予想される表現
18 行目:39 文字目:予想される表現
22 行目:3 文字目:予想される表現
23 行目:25 文字目:予想される表現
24 行目:41 文字目:予想される表現
28 行目:3 文字目:予想される表現
29 行目:3 文字目:予想される表現
30 行目:33 文字目:予想される表現

1.3.2.4 考察

a) 全角日本語を変数に指定した場合、うまく出力できなかった.
b) 全角文字は変数として扱えない事がわかった.

1.4 工夫…！

1.4.1 ソースコードの一部

```
9   #include <stdio.h>
10
11  int main()
12  {
13    int price = 1234, pay = 10000;
14    int balance, amount;
15    /******************************** scanf */
16    printf("How much is the price ?   => "); scanf("%d",&price);
17    printf("Do you pay how much ? => "); scanf("%d",&pay);
18    puts("----");
19    if(price > pay){
20      printf("ERROR!\n");
21    }
22    else{
23      /******************************** balance */
24      balance = pay - price;
25      printf("price = %dyen, pay = %dyen, change = %dyen\n",price,pay,balance);
26      puts("----");
27
28      /******************************** 5000円 */
29      amount  = balance / 5000;
30      balance = balance % 5000;
31      printf("5000-yen note = %d\n",amount);
32      price = amount;
33      /******************************** 2000円 */
34      amount  = balance / 2000;
35      balance = balance % 2000;
36      printf("2000-yen note = %d\n",amount);
37      price = price+amount;
38      /******************************** 1000円 */
39      amount  = balance / 1000;
40      balance = balance % 1000;
41      printf("1000-yen note = %d\n",amount);
42      price = price + amount;
43      /******************************** 500円 */
44      amount  = balance / 500;
45      balance = balance % 500;
46      printf("500-yen  coin = %d\n",amount);
47      pay = amount;
48      /******************************** 100円 */
49      amount  = balance / 100;
50      balance = balance % 100;
51      printf("100-yen  coin = %d\n",amount);
52      pay = pay + amount;
53      /******************************** 50円 */
54      amount  = balance / 50;
55      balance = balance % 50;
56      printf("50-yen   coin = %d\n",amount);
57      pay = pay + amount;
58      /******************************** 10円 */
59      amount  = balance / 10;
60      balance = balance % 10;
61      printf("10-yen   coin = %d\n",amount);
62      pay = pay + amount;
63      /******************************** 5円 */
64      amount  = balance / 5;
65      balance = balance % 5;
66      printf("5-yen    coin = %d\n",amount);
67      pay = pay + amount;
68      /******************************** 1円 */
69      printf("1-yen    coin = %d\n",balance);
70      pay = pay + balance;
```

```
71      printf("----\n");
72      printf("Number of note --> %d,Number of coin --> %d\n",price,pay);
73      price = price + pay;
74      printf("Number of note and coin --> %d\n",price);
75    }
76
77    return(0);
78  }
```

1.4.2 出力結果(1)　※変数 price、pay にはそれぞれ 1234、10000 を入力

```
How much is the price ?   => 1234
Do you pay how much ? => 10000
----
price = 1234yen, pay = 10000yen, change = 8766yen
----
5000-yen note = 1
2000-yen note = 1
1000-yen note = 1
500-yen coin   = 1
100-yen coin = 2
50-yen   coin = 1
10-yen   coin = 1
5-yen    coin = 1
1-yen    coin = 1
----
Number of note --> 3,Number of coin --> 7
Number of note and coin --> 10
```

1.4.3 出力結果(2)　※変数 price、pay にはそれぞれ 1234、999 を入力

```
How much is the price ?   => 1234
Do you pay how much ? => 999
----
ERROR!
```

1.4.4 解説

a) printf()関数によって対話的に対応.
b) お札と硬貨のそれぞれの合計枚数を最後に出力するようにした.
c) なるべく変数を多く使わないようにし、例題と同じく４つの変数を使用した。
d) レポートの課題範囲からは外れるが、計算結果にマイナスが現れるのを避けるために、if 文により変数 price に格納されている数が pay より多い場合にエラーを返すようにした.

2 int 型整数の下限・上限の値について、簡単なプログラムと実行結果を示し考察せよ。

2.1 テキスト PP.68 基数 16 の表記法を用いたプログラムを考えること。

2.1.1 ソースコードの一部

```
9   #include<stdio.h>
10
11  int main(){
12
13    int i ;
14
15    printf("Decimal number ==> "); scanf("%d",&i);
16    printf("(Dec):%010d\n",i);
17    printf("(Hex):0x%08x\n",i);
18
19    return(0);
20  }
```

2.1.2 出力結果　※ "i"には 734 を入力

```
Decimal number ==> 734
(Dec):0000000734
(Hex):0x000002de
```

2.1.3 方針

a) 10 進数の数字を入力し、10 進数と 16 進数に変換して表示するプログラムを作成した.
b) 16 進数を表す場合、「テキスト PP.68 基数 16 の表記法」を参考にし "0x"から始めるようにした.
c) 2 進数 32bit の数は、10 進数の場合 10 桁、16 進数の場合 8 桁で表されるので "%10d"、"%08x"により桁数を表し、何もない桁には 0 を詰めるために 0 を入力した.
d) 以降の考察では上記のプログラムを用いて、調べたい 10 進数の数字を入力し、10 進数と 16 進数で出力する.
e) テキスト P.409 によれば int 型整数の bit 数は 32bit であり、値域は 10 進法で "2147483647"～ "-2147483648"であるという.
f) それぞれの上限値、下限値とその前後の数字を入力して出力結果を観察する.

2.2 int 型整数の上限値について

2.2.1 出力結果　※変数 i には 2147483647 を入力

```
Decimal number ==> 2147483647
(Dec):2147483647
(Hex):0x7fffffff
```

2.2.2 考察

a) int 型整数の上限値である 2147483647 を入力した場合、10 進数と 16 進数共に問題なく表示された.
b) 次に、上限値の前後の数字を入力してみる.

2.3 上限値の前後について

2.3.1 出力結果(1)　※変数 i には 2147483648 を入力

```
Decimal number ==> 2147483648
(Dec):-2147483648
(Hex):0x80000000
```

2.3.2 出力結果(2)　※変数 i には 2147483646 を入力

```
Decimal number ==> 2147483646
(Dec):2147483646
(Hex):0x7ffffffe
```

2.3.3 考察

a) 上限値を超えた数を入力した場合、10 進数の出力にはマイナスがついて出力された（これは int 型整数の下限値である）.
b) 上限を超えない数を入力した場合、問題なく出力された.

2.4 int 型整数の下限値について

2.4.1 出力結果　※変数 i には-2147483648 を入力

```
Decimal number ==> -2147483648
(Dec):-2147483648
(Hex):0x80000000
```

2.4.2 考察

a) int 型整数の下限値である-2147483648 を入力した場合、10 進数は問題なく表示された.
b) １６進数には、2147483648 を入力した場合（2.3.1 項）と同じ結果が現れた。
c) 次に、下限値の前後の数字を入力してみる.

2.5 下限値の前後について

2.5.1 出力結果(1)　※変数 i には-2147483649 を入力

```
Decimal number ==> -2147483649
(Dec):2147483647
(Hex):0x7fffffff
```

2.5.2 出力結果(2)　※変数 i には-2147483647 を入力

```
Decimal number ==> -2147483647
(Dec):-2147483647
(Hex):0x80000001
```

2.5.3 考察

a) 下限値を超えた数を入力した場合、2.2.1 項の 10 進数、16 進数の出力と同じ結果が現れた.
b) 下限を超えない数を入力した場合、問題なく出力された.

2.6 0の前後の値について

2.6.1 出力結果(1)　※変数 i には 1 を入力

```
Decimal number ==> 1
(Dec):0000000001
(Hex):0x00000001
```

2.6.2 出力結果(2)　※変数 i には-1 を入力

```
Decimal number ==> -1
(Dec):-000000001
(Hex):0xffffffff
```

2.6.3 考察

a) 1 を入力した場合、１０進数、１６進数共に問題なく出力された.
b) -1 を入力した場合、１０進数では問題なく出力されたが、１６進数では "ffffffff" が出力された.

2.7 int 型整数の表現の考察

2.7.1 考察

a) 16 進数の出力（Hex）を観察する
b) "i=1"の場合で "0x00000001"であり最小、 "i=-1"の場合で "0xffffffff"であり最大になっている.
c) 上限値、下限値の場合では "0x7fffffff"、 "0x80000000"となり、隣同士の数として出力された.
d) しかし 10 進数の出力（Dec）を観察すると、上限値+1 を入力した時と下限値、下限値+(-1)を入力した時と上限値の値が等しくなっている。
e) コンピュータの内部数値は２進数を基本としているので、ある数字を 10 進数として入力した場合その数を２進数へ変換する.
f) この時、２進数で符号を表現するために「2 の補数」を用いると考えられる.
g) よって、32bit で２進数の先頭（32 桁目）が 1 になる場合は１０進法でマイナスとして扱う.

2.7.2 それぞれの基数の関係(32bit)

１０進数	２進数	１６進数
0	0000...0000	0x00000000
1	0000...0001	0x00000001
2	0000...0010	0x00000002
~		

2147483646	0111...1110	0x7ffffffe
2147483647(上限)	0111...1111	0x7fffffff
-2147483648(下限)	1000...0000	0x80000000
-2147483647	1000...0001	0x80000001
~		
-2	1111...1110	0xfffffffe
-1	1111...1111	0xffffffff

h) 前ページの表より、上限値、下限値を超えた値を扱う場合、２進数へ変換したときにその数は「２の補数表現」として扱うので、出力結果にマイナスが現れたりする.
i) 16 進数の出力の変化を見ると、前ページの表は上下の値が繋がり、1 順しているという解釈が分かりやすい.
j) 確認のため、32bit の２進数で表すことのできる最大の数（4294967296）を入力してみる.

2.7.8 出力結果　※ "i"には 4294967296 を入力

```
調べたい数を入力==>4294967296
(Dec):0000000000
(Hex):0x00000000
```

a) 予想通り、32bit の２進数を超えた数を入力すると、オーバーフローを起こしてそれぞれに０が出力された.
b) 以上より、int 型整数は 32bit の２進数として扱い、それは「2 の補数表現」で表すので上限値は 2147483647(=2^31-1)であり、下限値は-2147483648(=-2^31)となる.

3 エラーについて考察せよ。

3.1 方針

a) 1.3.2.2 項について、全角日本語を変数として扱うことができないことがわかった.
b) ここでは、様々な記号を用いて変数に扱うことができる文字を模索してみる.

3.2 #,$,%,&の記号について。

3.2.1 ソースコード全体

```
1   #include<stdio.h>
2   int main(){
3
4     int #=1,$=2,%=3,&=4;
5     int A;
6
7     A= # + $ + % + &;
8     printf("A=%d\n",A);
9
10    return(0);
11  }
```

3.2.2 コンパイル時のエラー

```
report@2-5.c:4:7: error: expected identifier or '('
  int #=1,$=2,%=3,&=4;
      ^
report@2-5.c:7:6: error: expected expression
  A= # + $ + % + &;
     ^
```

3.2.3 考察

a) printf 関数内では記号はうまく表示できないので、直接の表現は避けるようにした.
b) "#"についての箇所にエラーが表示された.
c) "#"の部分を消去し実行したが、今度は "$"の箇所でエラーが出力された.
d) 以下同様に操作をしたが、すべての記号についてエラーが出力された.
e) よって、記号は変数として扱うことができないことがわかった.

ーあとがきー

A:参考サイト・文献

・『ビット(bit)とバイト(Byte)　情報量の単位と 2 進数』
http://www.biwako.shiga-u.ac.jp/sensei/mnaka/ut/binarydigit.html
・『2 の補数を理解する (1)』
http://d.hatena.ne.jp/simply-k/20100824/1282743815
・『C 実践プログラミング 第三版』