プログラミングⅠ

report#8

提出日:7 月 18 日(木)

所属:工学部情報工学科

学籍番号:135713B

氏名:天願　寛之

目次

1. 入力した正の整数を降順に並べ換えて出力するプログラムを作成せよ。プログラムは個別にコンパイルし、make コマンドで実行すること。

入力データは50以下とし、以下の数が混在しているとする。 16進数：先頭1文字がxまたはX（エックスの小文字か大文字） 8進数：先頭1文字が0（零） 10進数：先頭1文字が0（零）以外の数字

考察ポイント：複数あるポインタ変数が何を指しているかの把握、Makeコマンドでの実行

プログラムソースコード

1-a. ソースcardinal.c

```
/*
  Program    : cardinal.c
  Comment    : 基数変換と整列処理
*/

#include <stdio.h>
#include <string.h>
#define  MAX 256

void conv10(char **x, int *k, int n);
void select_sort(int x[], int m[], int n);
void print_num(char *x[], int m[], int n);
void msg();

int main(){
  char *dt[50], num[10], buf[MAX], *p=buf;
  int  n=0, len, i10[50], move[50];

  puts("-------- Input");
  while(gets(num) != NULL) {
    len = strlen(num);
    if(p > buf+MAX-(len+1))  break;
    strcpy(p, num);
    dt[n] = p;
    p += strlen(num)+1;
    n++;
  }
  puts("-------- Result");
  conv10(dt,i10,n);
  select_sort(i10,move,n);
  print_num(dt,move,n);
  msg();

  return(0);
}
```

1-b. ソースconv10.c

```
/*
  Program    : conv10.c
*/

void conv10(char **x, int *k, int n){
  while(n-- > 0){
    switch(**x){
    case '0' :
      sscanf(*x+1, "%o", k);
      break;
    case 'x' :
    case 'X' :
      sscanf(*x+1, "%x", k);
      break;
    default :
      sscanf(*x, "%u", k);
      break;
    }
    x++;
    k++;
  }
}
```

1-c. ソースselect_sort.c

```
/*
  Program    : select_sort.c
*/

void select_sort(int x[], int m[], int n){
  int  i, j, k, w;

  for(i=0; i<n; i++)  m[i]=i;
  for(i=0; i<n-1; i++){
    k=i;
    for(j=i+1; j<n; j++)
      if( x[j] > x[k] )  k=j;
    w    = x[i];
    x[i] = x[k];
    x[k] = w;
    w    = m[i];
    m[i] = m[k];
    m[k] = w;
  }
}
```

1-d. ソースprint_num.c

```
/*
  Program    : print_num.c
*/

void print_num(char *x[], int m[], int n){
  int  i;

  for(i=0; i<n; i++){
    puts( x[m[i]] );
  }
}
```

1-e. ソースmsg.c

```
/*
  Program    : msg.c
*/
int msg(){
  printf("#### Message from C #### By ICHIRO¥n");
  return(0);
}
```

1-f. ソースmakefile

```
#
#cardinal.cのためのmakeファイル
#

cardinal: cardinal.o conv10.o select_sort.o print_num.o msg.o
	gcc -o cardinal cardinal.o conv10.o select_sort.o print_nu¥
m.o msg.o

cardinal.o: cardinal.c
	gcc -c cardinal.c

conv10.o: conv10.c
	gcc -c conv10.c

select_sort.o: select_sort.c
	gcc -c select_sort.c

print_num.o: print_num.c
	gcc -c print_num.c

msg.o: msg.c
	gcc -c msg.c
```

実行結果

```
-------- Input
x21
021
055
22
-------- Result
055
x21
22
021
#### Message from C #### By ICHIRO

[プロセスが完了しました]
```

1-a-1. ソースcardinalについての考察

1-a-2. ソースcardinal.c

```
01 /*
02   Program    : cardinal.c
03   Comment    : 基数変換と整列処理
04 */
05
06 #include <stdio.h>
07 #include <string.h>
08 #define  MAX 256 /*MAXを256と定義*/
09
10 void conv10(char **x, int *k, int n); /*関数conv10のプロトタイプ宣言*/
11 void select_sort(int x[], int m[], int n); /*関数select_sortのプロとタイプ宣言*/
12 void print_num(char *x[], int m[], int n); /*関数print_numのプロトタイプ宣言*/
13 void msg(); /*関数msgのプロトタイプ宣言*/
14
15 int main(){
16   /*char型のポインタ配列dt[50],配列num[10],buf[MAX],ポインタpを宣言.ポインタpに配列bufのアド¥
17 レスを代入*/
18   char *dt[50], num[10], buf[MAX], *p=buf;
19   /*int型の変数n,len,配列i10[50],move[50]を宣言.変数nに0を代入*/
20   int  n=0, len, i10[50], move[50];
21
22   puts("-------- Input");
23   /*whileループ*/
24   while(gets(num) != NULL) {
25     len = strlen(num); /*関数strlenでnumの文字列の長さ取得しlenに代入*/
26     if(p > buf+MAX-(len+1))  break; /*if文,p > buf+MAX-(len+1)であればループを抜ける*/
27     strcpy(p, num); /*関数strcpy,文字列pに文字列numをコピー*/
28     dt[n] = p; /*ポインタ配列dt[n]にpに代入された文字列の先頭アドレスを代入*/
29     p += strlen(num)+1;           /*p足す文字列numの長さ+1を取得しpに代入*/
30     n++; /*nのインクリメント*/
31   }
32   puts("-------- Result");
33   conv10(dt,i10,n); /*関数conv10,引数dt,i10,nを取り呼び出し*/
34   select_sort(i10,move,n); /*関数select_sort,引数i10,move,nを取り呼び出し*/
35   print_num(dt,move,n); /*関数print_nuk,引数dt,move,nを取り呼び出し*/
36   msg(); /*関数msg,引数なしで呼び出し*/
37
38   return(0);
39 }
```

1-a-3. 解析

イ. 08行目でMAXを256と定義している。これはNULLを含んだ文字数を256文字分が上限であるということである。

ロ. 10行目でvoid型関数conv10のプロトタイプ宣言している。

ハ. 11行目でvoid型関数select_sortのプロトタイプ宣言をしている。

ニ. 12行目でvoid型関数print_numのプロトタイプ宣言をしている。

ホ. 13行目でvoid型関数msgのプロトタイプ宣言をしている。

ヘ. 18行目でchar型のポインタ配列dt[50]、配列num[10]、buf[MAX]、ポインタpを宣言。ポインタpに配列bufの先頭アドレスを代入している。 <4>

ト. 20行目でint型の変数n、len、配列i10[50]、move[50]を宣言。変数nに0を代入している。

チ. 24行目から31行目でwhile文を用いてループしている。関数getsを用いて入力された行の値を配列numに代入し、その値がNULL以外ならループを続けている。

リ. 25行目で関数strlenを用いて配列numに代入された文字数を取得(このとき最後に追加されるNULLは含まない)し、int型変数lenに代入している。

ヌ. 26行目、if文で”p > buf+MAX-(len+1)”ならばプログラムを抜ける。”p”はbufの先頭アドレスを代入されているので値はbufの先頭アドレスである。”buf+MAX-(len+1)”について、bufは確保した配列の先頭アドレスの値、MAXは定義した通り256、lenは入力した文字数である。

ル. 27行目では、関数strcpyを用いてchar型のポインタ変数pに文字列numをコピー(このときNULLも含んでいる)している。よって、間接的にbufには入力した文字+NULLが入る。

ヲ. 28行目では、char型ポインタ配列dtのn番目の配列にpの指すアドレスを代入。

ワ. 29行目では、pにp+入力した文字数+1の値を代入している。pは最初bufの先頭アドレスを持つので最初は先頭アドレスから入力した文字数+1分を移動したアドレスが代入されたことになる。

カ. 30行目では、int型変数nのインクリメントを行っている。このインクリメントにより1回目のループで28行目での配列dt[1]に29行目で変わったpの指すアドレスを代入することになる。

ヨ. 33行目では、関数conv10を引数にdt、i10、nを取り呼び出している。

タ. 34行目では、関数select_sortを引数にi10、move、nを取り呼び出している。

レ. 35行目では、関数print_numを引数にdt、move、nを取り呼び出している。

ソ. 36行目では、関数msgを引数なしで呼び出している。

1-a-4. 考察

イ. 26行目のif文での動作についてMAXの定義を変更し、いくら文字数を入力をしたらループを抜けるのか試してみる。(MAXの定義を5に変える)

1-a-4-イ-1. 出力結果(入力は12345とする)

```
-------- Input
12345
-------- Result
#### Message from C #### By ICHIRO

[プロセスが完了しました]
```

入力した文字数はMAXを超えていないがbreakしたことが分かる。 <5>

1-a-4-イ-2. 出力結果(入力は1234とする)

```
-------- Input
1234
p = 1497303433 buf = 1497303433 MAX = 5 len = 4
-------- Result
1234
#### Message from C #### By ICHIRO

[プロセスが完了しました]
```

演算を分かりやすく理解するため26行目の後に下記のような1行を書き加えている。

```
printf("p = %d buf = %d MAX = %d len = %d¥n",p,buf,MAX,len);
```

出力結果より4文字の入力によって26行目の左辺と右辺の式は等号を示す事になり、while文でのループを抜けResultを出し、breakしていないことが分かる。

1-a-4-イ-3. 出力結果(入力は1,12,123,1234と異なる行で文字数を増やして入力する)

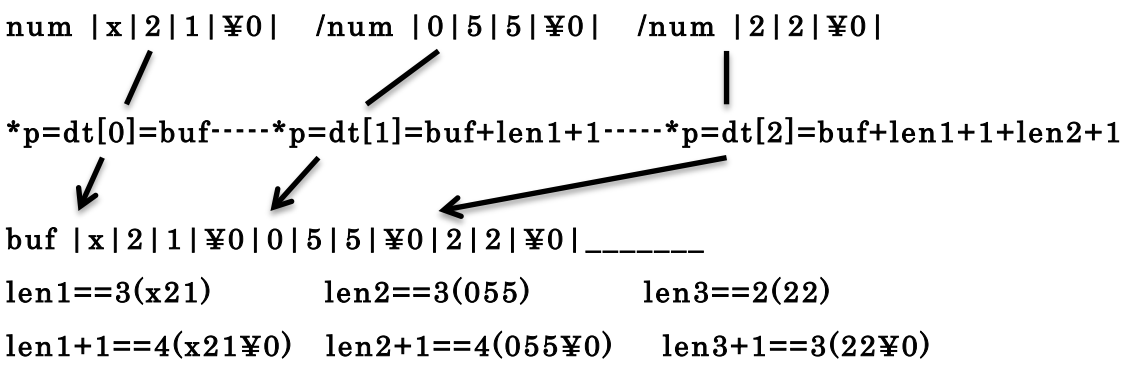
```
-------- Input
1
p = 1530423438 buf = 1530423438 MAX = 256 len = 1
12
p = 1530423440 buf = 1530423438 MAX = 256 len = 2
123
p = 1530423443 buf = 1530423438 MAX = 256 len = 3
1234
p = 1530423447 buf = 1530423438 MAX = 256 len = 4
-------- Result
1234
123
12
1
#### Message from C #### By ICHIRO

[プロセスが完了しました]
```

出力結果より、ポインタpの持つ値は入力した文字数+1分ずれていることが分かる。この+1分はNULLが入ることが分かる。これらの結果より文字を入力していくにつれポインタpの指すアドレスはその文字数+1分埋まっていくので最大で入力出来る文字数はNULLを含んだ256文字だと分かる。

ロ. 1-a-4の考察イ.より29行目でpに代入した”p+文字数+1”の+1はNULL文字のことであり、28行目でdt[n](n = 0,1,2…)に代入されているのは入力していった値のそれぞれの先頭アドレスだと分かる。

ハ. 図・イメージ(入力した文字列を3つ「x21、055、22」と考える。



1-b-1. ソースconv10についての考察

1-b-2. ソースconv10.c

```
01 /*
02   Program    : conv10.c
03 */
04
05 void conv10(char **x, int *k, int n){
06   while(n-- > 0){
07     switch(**x){
08     case '0' :
09       sscanf(*x+1, "%o", k);
10       break;
11     case 'x' :
12     case 'X' :
13       sscanf(*x+1, "%x", k);
14       break;
15     default :
16       sscanf(*x, "%u", k);
17       break;
18     }
19     x++;
20     k++;
21   }
22 }
```

1-b-3. 解析

イ. 05行目のvoid型関数conv10でmain関数から受け取った引数は”char **x”が”dt”、”int *k”が”i10”、”int n”が”n”である。

ロ. 06行目はwhile文を用いて変数nが0より大きい間デクリメントしながら07行目から21行目までをループする。入力した文字列が3つなら”cardinal.c”によりn=3の値が与えられているので、3回だけループする事ができる。

ハ. 07行目からのswitch-case文について。

```
01 switch (c) {
02         case  A: 処理 A;
03             break;
04         case  B: 処理 B;
05             break;
06         case  C:処理 C;
07             break;
08         default:処理 N;
```

上記のようなswitch-case文では、01行目の”c”の値が”A”のとき処理Aが行われる。以下同様で、08行目の”default”はオプションであり”無くても良い”という意味と同義である。つまり、caseで分けた他の条件以外ということである。ソースconv10.cでは**xの値、つまりdt[n]の指す文字列の最初のアドレスが持つ要素が’0’か’x’または’X’か、それ以外で処理を分岐している。

ニ. 08行目から10行目について、case ‘0’、つまり入力した文字列、または文字の最初の文字が0であれば関数sscanfにより”*x+1”つまりdt[n]の指すアドレスを１つずらして、ずらした先のアドレスがもつ要素の書式を8進数に換えてその値をポインタ変数kに代入している。

ホ. 11行目から13行目について、case’x’または’X’、つまり入力した文字列、または文字の最初の文字がxまたはXであれば関数sscanfにより”*x+1”つまりdt[n]の指すアドレスを1つずらして、ずらした先のアドレスがもつ要素の書式を16進数に換えてその値をポインタ変数kに代入している。

ヘ. 15行目から17行目について、caseで分けた他の条件以外のときは関数sscanfにより*x*つまりdt[n]の指すアドレスがもつ要素の書式を符号なし10進数に換えてその値をポインタ変数kに代入している。

1-b-4. 考察(入力した文字列を3つ「x21、055、22」と考える。)

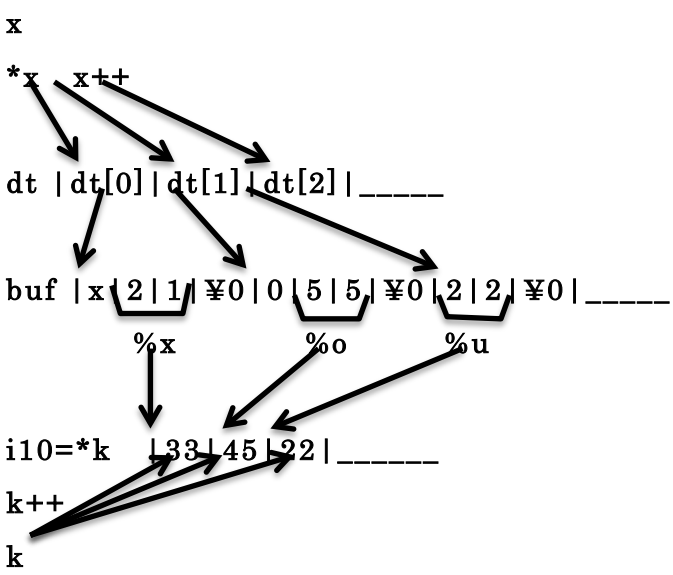
イ. 実際の出力結果

```
-------- Input
x21
055
22
-------- Result
case'x' *x+1 = 21
case'x' **x = x
case'0' *x+1 = 55
case'0' **x = 0
case'def' *x = 22
case'def' **x = 2
055
x21
22
#### Message from C #### By ICHIRO

[プロセスが完了しました]
```

この出力結果より関数sscanfにより変換されるのは、それぞれ先頭が'x'または'X'、'0'であればその次からの値である。またそれ以外の条件に当てはまる値はその先頭の値からであることが分かる。

ロ. 図・イメージ



1-c-1. ソースselect_sort.cについての考察

1-c-2. ソースselect_sort.c

```
01 /*
02   Program    : select_sort.c
03 */
04
05 void select_sort(int x[], int m[], int n){
06   int  i, j, k, w;
07
08   for(i=0; i<n; i++)  m[i]=i;
09   for(i=0; i<n-1; i++){
10     k=i;
11     for(j=i+1; j<n; j++)
12       if( x[j] > x[k] )  k=j;
13     w    = x[i];
14     x[i] = x[k];
15     x[k] = w;
16     w    = m[i];
17     m[i] = m[k];
18     m[k] = w;
19   }
20 }
```

1-c-3. 解析

イ. 05行目でvoid型関数select_sortがmain関数から受け取った引数は”int x[]”が”i10”、”int m[]”が”move”、”int n”が”n”である。

ロ. 06行目でint型変数i、j、k、wを宣言している。

ハ. 08行目でfor文で変数iの初期値は0、変数iが引数で受け取った変数nより小さい間ループを続け、ループ毎に変数iをインクリメントしている。ループしている間変数配列m[i]に変数iを代入している。

ニ. 09行目のfor文で変数iの初期値は0、変数iが引数で受け取った変数n-1より小さい間ループを続け、ループ毎に変数iをインクリメントしている。10行目ではループしている間変数kに変数iを代入し、また11行目for文で変数jの初期値を変数i+1にし、変数jが引数で受け取った変数nより小さい間ループを続け、ループ毎に変数jをインクリメントしている。11行目でのforループは12行目でのif文により”x[j] > x[k]”ならば変数kに変数jを代入するという作業をし、それを抜けると13行目で”w”に”x[i]”を代入。14行目で”x[i]”に”x[k]”を代入。15行目で”x[k]”に”w”を代入。16行目で”w”に”m[i]”を代入。17行目で”m[i]”に”m[k]”を代入。18行目で”m[k]”に”w”を代入している。

1-c-4. 考察(入力した文字列を3つ「x21、055、22」と考える。)

イ. 入力した文字列が3つの場合、main関数で変数nは3を代入されているので08行目のfor文では、m[0]=0、m[1]=1、m[2]=2と代入される。

ロ. 09行目のfor文での条件式は"i<2"となるのでループは1回となる。

ハ. 10行目の変数kには1回目の動作では0が、1回目のループでは1が代入される。

ニ. 11行目のfor文では1回目の動作で初期値には変数iが0なので1が代入される。また条件式とループ毎の動作によりループは1回となる。09行目のfor文による1回目のループのときはiが1なのでループはしない。以下選択法と細かい流れを示す。
その前にsscanで変換された値をint型配列i10で受け取るにあたり、変換された値を求めた結果以下の通りだと分かった。

x21=33, 055=45, 22=22

実際に流れの結果を示しておく。

```
-------- Input
x21
p = 1424619662 buf = 1424619662 MAX = 256 len = 3
055
p = 1424619666 buf = 1424619662 MAX = 256 len = 3
22
p = 1424619670 buf = 1424619662 MAX = 256 len = 2
-------- Result
m[0]==1,m[1]==0,m[2]==2,x[0]==45,x[1]==33,x[2]==22
055
x21
22
#### Message from C #### By ICHIRO

[プロセスが完了しました]
```

選択法

i=0　x[0]=33-x[1]=45-x[2]=22　　配列m(move)　m[0]=0-m[1]=1-m[2]=2

33と45を比較

22と45を比較

----------配列xと配列mで各々中身を入れ替え

i=1　x[0]=45-x[1]=33-x[2]=22　　m[0]=1-m[1]=0-m[2]=2

33と22を比較__

----------何もしない

最終結果_____ x[0]=45-x[1]=33-x[2]=22 / m[0]=1-m[1]=0-m[2]=2

x[0]=33,x[1]=45,x[2]=22

m[0]=0,m[1]=1,m[2]=2.

09行for文による1回目の動作

i=0,k=0

11行for文による1回目の動作

j=1

if文

x[1] > x[0]　つまり　45 > 33は真

k=1

11行for文による2回目の動作

j=2

if文

x[2] > x[1]　つまり　22 > 45は偽

w=x[0]……w=33

x[0]=x[1]……x[0]=45

x[1]=w……x[1]=33

w=m[0]……..w=0

m[0]=m[1]…..m[0]=1

m[1]=w……m[1]=0

09行for文による2回目の動作

i=1,k=1

11行for文による1回目の動作

j=2

if文

x[2] > x[1]　つまり　22 > 33は偽

w=x[1]……w=33

x[1]=x[1]……x[1]=33

x[1]=w……x[1]=33

w=m[1]……..w=0

m[1]=m[1]….m[1]=0

m[1]=w……..m[1]=0

結果m[0]==1,m[1]==0,m[2]==2,x[0]==45,x[1]==33,x[2]==22

1-c-4考察イで先に出力した結果と同じになった。　<12>

1-d-1. ソースprint_num.cについての考察

1-d-2. ソースprint_num.c

```
01 /*
02   Program    : print_num.c
03 */
04
05 void print_num(char *x[], int m[], int n){
06   int  i;
07
08   for(i=0; i<n; i++){
09     puts( x[m[i]] );
10   }
11 }
```

1-d-3. 解析

イ. 05行目でvoid型関数print_numがmain関数から受け取った引数は、”char *x[]”が”dt”、”int m[]”が”move”、”int n”が”n”である。

ロ. 06行目でint型変数iを宣言している。

ハ. 08行目のfor文は変数iの初期値0、変数iが変数nより小さい間ループ、ループ毎に変数iをインクリメントする。

ニ. 09行目のputsで、ループの間x[m[i]]を出力し続ける。

1-d-4. 考察(入力した文字列を3つ「x21、055、22」と考える。)

イ. x[]は配列の先頭アドレス、m[]は配列の値である。

ロ. x[m[i]]はi=0のときm[0]==1、よってx[1]つまりdt[1]が出力される。関数convでdt[1]は”055”を指しているので”055”が出力される。

ハ. x[m[i]]はi=1のときm[1]==0、よってx[0]つまりdt[0]が出力される。関数convでdt[0]は”x21”を指しているので”x21”が出力される。

ニ. x[m[i]]はi=2のときm[2]==2、よってx[2]つまりdt[2]が出力される。関数convでdt[2]は”22”を指しているので”22”が出力される。

1-e-1. ソースmsg.cについての考察

1-e-2. ソースmsg.c

```
/*
  Program    : msg.c
*/
int msg(){
  printf("#### Message from C #### By ICHIRO¥n");
  return(0);
}
```

1-e-3. 解析

イ．関数printfで”#### Message from C #### By ICHIRO”を出力している。

1-e-4. 考察

イ．しなくていいんじゃないか？

2. 感想

流れを知って実際にどんな値を持つか調べるのに苦労しました。%uや%oの出力が結局うまく出来ずに別のところで値を確かめることになりました。ここまできても未だにうまくいかない事が多いので反省し、これからに生かしたいです。