プログラミング第 5 回の課題

Report#5(文字コード変換)

1. ライブラリ関数 islower(), toupper()を使い、下記のtrlowupプログラム を書き換えて、新規にtrupperプログラムを作成せよ。

今から課題をやっていくわけだが、課題が文章の意味が何の事だかさっぱり分からない。ということで、まずは分からない用語を調べ、課題の文章を分かりやすく訳す、という方法でやっていく。

分からない用語

ライブラリ関数：コンパイラメーカがよく使う機能をオブジェクトライブラリとして提供してくれるものを「標準ライブラリ関数」と言う。 入出力、文字列処理、文字処理、数学処理など機能別に多くのライブラリ関数が存在。

islower()　：小文字テスト。if文の書きかたを忘れたらこれを使うべし。
toupper()　：大文字テスト。if文の書きかたを忘れたらこれを使うべし。

ソースプログラム

```
/*
  Program  : trlowup.c
  Comments : translate lower case characters into upper case ones.
*/

#include <stdio.h>

char trlowup(char);

int main(){
  char c;

  while( (c=getchar()) != EOF )
    putchar( trlowup(c) );
  return(0);
}

char trlowup( char c ){
  if ( 'a' <= c && c <= 'z' )
     return( c-'a'+'A' );
  else
     return( c );
}
```

出力結果：

```
Ac Bb Cc Dd Ee Ff   Xx Yy Zz
AC BB CC DD EE FF   XX YY ZZ
```

ソースプログラムと分からない用語の意味を含め、課題の意味を分かりやすいようになおしてみる。
要するにこういう事だ。

trlowup.cのプログラムに含まれているif文がある。このif文を使わずにislowerと
touppperを使ってtrlowupを書き換えてtrupperというプログラムの名前にしなさい

trupperプログラム

```
  /*
    Program     : trupper.c
    Student-ID  : e075739A
    Author      : TSUHAKO Masaki
    Update      : 2007/6/14
    Comments    : translate lower case characters into upper case ones.
  */

  /*ヘッダー*/
  #include <stdio.h>
  #include <ctype.h>

 /*文字型の宣言*/
  char trupper(char);

  /*変数宣言*/
  int main(){
    char c;

  /*while文  EOF(End Of File)まできたら終了*/
   while( (c=getchar()) != EOF )
     putchar( trupper(c) );
   return(0);
 }

 /*cが小文字ならば大文字に変換*/
 char trupper( char c ){
   if (islower(c))
     return( toupper(c) ) ;
   else
     return( c );
 }
```

出力結果：

```
Aa Bb Cc Dd Ee Ff    Xx Yy Zz
AA BB CC DD EE FF    XX YY ZZ
```

trupperプログラムは基本的にあまりソースプログラムと変わっていない。変わった点は２つ。
i.ヘッダーに #include <ctype.h> を加えた
ii ifの（ ）内の文章を変えた

```
.      if ( 'a' <= c && c <= 'z' )
           return( c-'a'+'A' );
         else
           return( c );
```

↓

```
       if (islower(c))
         return( toupper(c) ) ;
       else
         return( c );
```

この結果によりif文の条件式の書き方を忘れた時、islowerとtoupperを使うことによって同じ動作のプログラムを作ることができる。

2.trupperプログラムを書き換えて、rot13暗号化プログラム、rot13復号化プログラムを作成せよ。
rot13とは、次のようにアルファベットを13文字ずらす暗号化の方法である。

```
A --> N     a --> n
B --> O     b --> o
C --> P     c --> p
....        ....
Z --> M     z --> m
```

分からない用語

rot13暗号化プログラム：アルファベットを一文字毎に13文字後のアルファベットに置き換える。
AはNに、BはOに置き換えられる。（上にもかかれている）

ソースプログラムは1番で作ったtrupperプログラムである。

rot13暗号化プログラム1

```
/*ヘッダー*/
  #include <stdio.h>
  #include <ctype.h>

/*文字型の宣言*/
char trupper(char);

/*変数宣言*/
int main(){
  char c;

printf("すべて大文字にし,rot13方式で暗号化\n");
  /*while文  EOF(End Of File)まできたら終了*/
  while( (c=getchar()) != EOF )
    putchar( trupper(c) );
  return(0);
}

/*cが小文字ならば大文字に変換*/
char trupper( char c ){

  if (islower(c))
    return( toupper(c)+13 ) ;
  else
    return( c );
}
```

出力結果：

```
すべて大文字にし,rot13方式で暗号化
abcdefghijklmnopqrstuvwxyz
NOPQRSTUVWXYZ[\]^_`abcdefg
```

rot13という暗号化プログラムはASCIIコードを10進数にした時の数字が13ずれている、ということをヒントに上のプログラムを作ってみた。その結果、a(A)～m(M)の暗号化はうまくいった。しかしn(N)～z(Z)、つまり、ASCIIコードで10進数表現になおした時、＋13をすると記号になってしまうものがうまくいかなかった。最初の半分は成功だが、残り後半の半分が失敗という結果になった。
どうすればいいか。
A～M、またはa～mはうまくいく。しかし、後半の13個のアルファベットがうまくいかない。
N(n)とA(a)をASCIIコードの10進数で表してみる。

	A	N	a	n
10進数	65	78	97	110

ここで問題になっていることがN→A、n→aへの変換だ。考えられる方法は演算子の工夫である。何をどう使用するか、ということなのだがそのやり方がわからない。なのでまずは情報を整理することにする。

情 報

i　ASCIIコードでいうA～Z、a～zは10進数で65～90、97～122である。
ii　アルファベットは26文字である。
iii 演算子には、乗算、除算、加算、減算、剰余がある。

以上のことを参考にし、rot13暗号化を考える。まず前ページのrot13暗号化プログラムでは演算子の『＋』を使った。ということは、もう一つの演算子『－』を使ったif文を作れば解決である
しかしもう一つの方法がある。演算子『％』剰余を使用した方法だ。

『％』を使った実験：
あるアルファベットを？とする。その？をアルファベットの総数26で割ってみる。すると？＝Aの場合、あまりは13。？＝Nの場合、あまりは0。ここで気になったのが、Aのあまり13という数字である。これはrot13に使えるのではないか。そこでA～Zの剰余を調べてみる。
26で割ったらどうなるか：

アルファベット	剰余	アルファベット	剰余
A(u)	13	N(h)	0
B(v)	14	O(i)	1
C(w)	15	P(j)	2
D(x)	16	Q(k)	3
E(y)	17	R(l)	4
F(z)	18	S(m)	5
G(a)	19	T(n)	6
H(b)	20	U(0)	7
I(c)	21	V(p)	8
J(d)	22	W(q)	9
K(e)	23	Z(r)	10
L(f)	24	Y(s)	11
M(g)	25	Z(t)	12

上の表を見てみるとちょうど13ずつずれている。NのASCIIコード10進数表現は78。Aは65。今、問題となっていたことは、『N→A』にすることである。なので、上の表の結果から、？というアルファ

ベットを26で割り、その剰余に+65（小文字の場合は+97）をすれば13の順番がずれたアルファベットになる。

rot13暗号化プログラム2

```
/*ヘッダー*/
  #include <stdio.h>
  #include <ctype.h>

/*文字型の宣言*/
char trupper(char);

/*変数宣言*/
int main(){
  char c;

  printf("大文字は小文字,小文字は大文字にし、rot13方式で暗号化\n");
  /*while文  EOF(End Of File)まできたら終了*/
  while( (c=getchar()) != EOF )
    putchar( trupper(c) );
  return(0);
}

/*cが小文字ならば大文字に変換*/
char trupper( char c ){


      if (islower(c))
           return(toupper(c)%26+65) ;
      /*cが大文字ならばそのまま変換*/
      else if (isupper(c))
        return(c%26+97);
 return(c);
     }
```

出力結果：

```
大文字は小文字,小文字は大文字にし、rot13方式で暗号化
abc xyz ABC XYZ 123 $%&
NOP KLM nop klm 123 $%&
NOP KLM nop klm 123 $%&
abc xyz ABC XYZ 123 $%&
```

成功である。

このrot13暗号化プログラミングは、2つの方法がある。
i　if文を2つくり加算『＋』、減算『―』を使って暗号化する。

ii 『%』の剰余の性質を利用して暗号化する。

%を使った場合、加算する数字が65と97だった。これは『A』と『a』で代用できる。
『A』と『a』で代用できるのか実験：

```
      if (islower(c))
           return(toupper(c)%26+'A') ;
      /*cが大文字ならばそのまま変換*/
      else if (isupper(c))
        return(c%26+'a');
 return(c);
     }
```

出力結果：

```
大文字は小文字,小文字は大文字にし、rot13方式で暗号化
abc xyz ABC XYZ 123 $%&
NOP KLM nop klm 123 $%&
NOP KLM nop klm 123 $%&
abc xyz ABC XYZ 123 $%&
```

できた。

2. オリジナルの暗号化·復号化

今までの知識を生かし、オリジナルな暗号化·復号化を作ってみる。

まず考えたことは、rot13を利用した暗号化·復号化だ。rot13方式の特徴は、ずばり『サイクル(輪)』である。A~Z（a~z）のアルファベットは全部で26。13個ずつズラすと元に戻る（サイクルする）。

この『サイクル』使ってオリジナルの暗号化·復号化を作る。

構想

まずアスキーコードでの10進数を元に、アルファベット、数字、記号の個数を数えてみる

アルファベット	英大文字、英小文字ともに26個ずつ
数字	0~9までの計10個
記号	計26個

興味深いことに、記号とアルファベットの数が同じである。これを利用する。

rot13では、アルファベット26個を13個ずつ分けていた。なので記号も2つのグループに分けてみる（数字は10個しかなかったので省く）

	英大文字	記号	英小文字
パターン1	A ~ M（65~77）	SP ~ ,（32~44）	a~m（97~109）
パターン2	N ~ Z（78~90）	- ~ /（45~47）[~ ‘（91~96） { ~ ~（123~126）	o~z（111~122）

（カッコ内はASCIIコード10進数表示）

パターン2の記号は3つのグループがある。

このパターン1とパターン2の英大文字、記号、英小文字のサイクルを作る。

あとは、rot13暗号化プログラム2の中身を少し変更すればいい。

```
/*ヘッダー*/
  #include <stdio.h>
  #include <ctype.h>

/*文字型の宣言*/
char trupper(char);

/*変数宣言*/
int main(){
  char c;

  printf(" サイクル式暗号化・復号化\n");
  /*while文  EOF(End Of File)まできたら終了*/
  while( (c=getchar()) != EOF )
    putchar( trupper(c) );
  return(0);
}

/************************/
char trupper( char c ){

/*英大の変換 A~M*/
if ('A'<=c&&c<='M')
  return(c+32) ;
/*英大の変換 N~Z*/
else if ('N'<=c&&c<='Z')
  return(c-46);
/*英小の変換 a~m*/
else if ('a'<=c&&c<='m')
  return(c-19);
/*英小の変換 n~z*/
else if ('n'<=c&&c<='z')
  return(c-45);
/*特定の記号の変換 ASCIIコード10進数32~44*/
else if (' '<=c&&c<=',')
  return(c+78);
/*それ以外はそのまま*/
return(c);
}
```

実行結果：

	サイクル式暗号化•復号化
1	inNZnFGHQLVATn"EBTENZZVAT.&VZRnVFn23:00nBAn)RQARFQNL.hHEELnHCooniGnVFnQNATREBHF.
2	VA ,Afgh#l(a&Apeb&e ,,(a&.t(,$A(fA23:00AbaAw$#a$f# l.UheelAhcBBAVgA(fA# a&$ebhf.
3	
4	
5	VA ,Afgh#l(a&Apeb&e ,,(a&.t(,$A(fA23:00AbaAw$#a$f# l.UheelAhcBBAVgA(fA# a&$ebhf.
6	(anzaSTUqYvNtaCROtRnzzvNt.GvzravSa23:00aONaJrqNrSqnY.'URRYaUPbba(TavSaqnNtrROUS.
7	
8	
9	(anzaSTUqYvNtaCROtRnzzvNt.GvzravSa23:00aONaJrqNrSqnY.'URRYaUPbba(TavSaqnNtrROUS.
10	vNAMN%&'D+I GNc$!G$AMMI G.gIMENI%N23:00N! NjED E%DA+.u'$$+N'"OONv&NI%NDA GE$!'%.
11	
12	
13	vNAMN%&'D+I GNc$!G$AMMI G.gIMENI%N23:00N! NjED E%DA+.u'$$+N'"OONv&NI%NDA GE$!'%.
14	I am studying Programming.Time is 23:00 on Wednesday.Hurry up!! It is dangerous.

成功だ。
このプログラムの売りは 4 回の復号化が必要ということ。（暗号化をいれたら 1 サイクルは 5 回必要）
上の出力結果は分かりやすくするため間をとったが、もう少し分かりやすく言うとこうだ。

行数 1 の文 → 行数 2 の文 → 行数 6 の文 → 行数 10 の文 → 行数 14 の文 → 行数 1 の文

よってオリジナルの暗号化･復号化プログラムの完成

感想：
今回の課題は、めちゃくちゃ楽しかった。暗号化って奥が深いなぁ～とも思った。
一番楽しかった（嬉しかった）のがオリジナルの暗号化復号化プログラムを作った時に、ちゃんと復号化ができたことだ。サイクルの方法は考えていたが、まさかこの方法でうまくいくとは…。
残念だったのが､(またオリジナルの暗号化･h復号化の話だが)ASCII コード 10 進数表現でいう 45~47 番、91~96 番、123~126 番の記号と数字の暗号化ができなかったことだ。サイクルの輪に入れようとしたが、個数や番号が中途半端で輪の中にいれることができなかった（やろうとしたらできると思うが、それではきれいなサイクルの輪にならない…ので断念)。
もっと効率のいい？というか、きれいな暗号化･復号化のプログラムを作ってみたい。

参考文献
先輩ズ
http://www.ie.u-ryukyu.ac.jp/~e065745/
http://www.ie.u-ryukyu.ac.jp/~e065762/
http://www.ie.u-ryukyu.ac.jp/~e065701/
http://www.ie.u-ryukyu.ac.jp/~e065702/http://www9.plala.or.jp/sgwr-t/c/sec07.html