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。

＊ サンプルプログラム

```
 1  /*
 2    Program  : trlowup.c
 3    Comments : translate lower case characters into upper case ones.
 4  */
 5
 6  #include <stdio.h>
 7
 8  char trlowup(char);
 9
10  int main(){
11      char c;
12
13      while( (c=getchar()) != EOF )
14        putchar( trlowup(c) );
15        return(0);
16  }
17
18  char trlowup( char c ){
19      if ( 'a' <= c && c <= 'z' )
20         return( c-'a'+'A' );
21      else
22         return( c );
23  }
```

＊ 実行結果

```
pw010% cat data1
abcdEFGhijKLmNoPqRstuVwXyZ

pw010% cc trlowup.c -o trlowup

pw010% trlowup
abcDEF
ABCDEF
xyzXyYyZz
XYZXYYYZZ
^C
pw010%

pw010% trlowup < data1
ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

＊ ライブラリ関数 islower(),toupper()を使ったプログラム

```
 1  /*
 2     Program    : trupper.c
 3     Student-ID : 095739K
 4     Author     : TOUME, Kouta
 5     Comments   : Used Library function islower() & toupper()
 6  */
 7
 8  #include<stdio.h>
 9  #include<ctype.h>
10
11  char trupper(char);
12
13  int main(){
14    char c;
15
16    while((c = getchar()) != EOF)
17      putchar(trupper(c));
18
19    return(0);
20  }
21
22  char trupper(char c){
23    if(islower(c))
24      return(toupper(c));
25    else
26      return(c);
27  }
```

＊ 実行結果

```
new-host:Report4 kouta$ cat data1
abcDEfghIjkLMNopqRstuVWXYz

new-host:Report4 kouta$ ./trupper < data1
ABCDEFGHIJKLMNOPQRSTUVWXYZ

new-host:Report4 kouta$ ./trupper
abcdef
ABCDEF
xYz#%&'
XYZ#%&'
^C
```

＊ 考　察

islower()関数
- 引数として渡した文字がアルファベットの小文字なら真、それ以外は偽になる関数である。

toupper()関数
- 引数として渡した文字がアルファベット小文字ならば、大文字にして値を返す関数である。

9行目
- islower()、toupper()関数を使うためには、ctype.h というヘッダーファイルを読み込む。

11行目
- trupper()をchar型の関数として定義。引数はchar型。

16-17行目
- getchar()で文字を1文字ずつ読み込んで、char型変数 c に代入。
- 変数 c の値が EOF と等しくなかったらputchar()で、trupper()関数に1文字ずつ値を渡す。

22-27行目
- char型関数trupper()は、char型の引数を受け取り変数 c に代入。
- if文。islower()関数 … 変数 c が小文字なら「真」を返す。そうでなければ「偽」を返す。
  真の場合 … toupper()関数 … 小文字を大文字にした値をmein関数に返す。
  偽の場合 … 大文字なので、そのままの値をmain関数に返す。

実行結果より、小文字以外の文字(大文字や記号)は変換されずそのまま出力されている。
また、while()文の終了条件である EOF は 「 ^C (controlキー + C)」であることが分かる。

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
```

＊ プログラム rot13.c

```
 1  #include<stdio.h>
 2  #include<ctype.h>
 3
 4  char rot13(char);
 5
 6  int main(){
 7    char c;
 8
 9    while((c = getchar()) != EOF)
10      putchar(rot13(c));
11
12    return(0);
13  }
14
15  char rot13(char c){
16    if(islower(c))
17      return((c - 'a' + 13) % 26 + 'a');
18    else if(isupper(c))
19      return((c - 'A' + 13) % 26 + 'A');
20    else
21      return(c);
22  }
```

＊実行結果

```
new-host:Report4 kouta$ cat data2
This is a pen.

new-host:Report4 kouta$ ./rot13 < data2
Guvf vf n cra.

new-host:Report4 kouta$ ./rot13
abcdefghijklm
nopqrstuvwxyz

ABCDEFGHILKLM
NOPQRSTUVYXYZ
^C
```

＊考　察

isupper()関数
- 引数として渡した文字がアルファベットの小文字なら真、それ以外は偽になる関数である。

4行目
- rot13()をchar型の関数として定義する。引数はchar型。

6-13行目
- trupperプログラムの13-20行目と同じなので省略する。

15-22行目
- char型関数 rot13() は、char型の引数を受け取り変数 c に代入。
- if文。islower()関数 … 変数 c が小文字なら「真」を返す。そうでなければ「偽」を返す。
- 真の場合

```
(c - 'a' + 13) % 26 + 'a'
```

変数 c から文字 'a' を引くことで、0 ～ 25までの数字を表現することができ、その値に 13 を足し、26 の剰余をとる。その値を 'a' に足すことで、a を n に、b を o に、… 、z を m に暗号化することができる。

- else-if文。isupper()関数 … 変数 c が大文字なら「真」を返す。
- 真の場合

```
(c - 'A' + 13) % 26 + 'A'
```

小文字と同様の処理をすることで、A を N に、B を O に、… 、Z を M に暗号化する。

- 偽の場合 … 小文字でも大文字でもない文字(記号)はそのまま出力される。

a を暗号化した場合、n になり、n を暗号化すると a に戻る。
同様に、b は o に、o は bとなるので、このプログラムは、暗号化だけでなく、復号化も兼ね備えたプログラムだということがわかる。

3. オリジナルの暗号化・復号化プログラムを作成せよ。

＊ 暗号化プログラム

```
 1  #include<stdio.h>
 2
 3  #define SHIFT 8
 4
 5  char encryption(char);
 6
 7  int main(){
 8    char c;
 9
10    while((c = getchar()) != EOF)
11      putchar(encryption(c));
12
13    return(0);
14  }
15
16  char encryption(char c){
17    if(c + SHIFT <= '~')
18      return(c + SHIFT);
19    else if(c + SHIFT >= '~')
20      return(('~' - c + SHIFT)+ ' ');
21    else
22      return(c);
23  }
```

＊ 復号化プログラム

```
16  char decryption(char c){
17    if(c - SHIFT >= ' ')
18      return(c - SHIFT);
19    else if(c - SHIFT <= ' ')
20      return('~' - (' ' - c + SHIFT));
21    else
22      return(c);
23  }
```

＊ 実行結果

```
＊ 暗号化
new-host:Report4 kouta$ ./Encryption
This is a pen.
\pq{(q{(i(xmv6^C

＊ 復号化
new-host:Report4 kouta$ ./Decryption
\pq{(q{(i(xmv6
This is a pen.^C
```

＊考　察

3行目
- #define文。SHIFT を 8 に置き換える。

5行目
- encryption()をchar型の関数として定義する。引数はchar型。

7-14行目
- trupperプログラムの13-20行目と同じなので省略する。

【暗号化の場合】
16-23行目
- char型関数 encryption() は、char型の引数を受け取り変数 c に代入。
- if文。
　　条件文　c + SHIFT <= '~'

　変数 c に SHIFT(8) を足した値が ASCIIコードの表示可能文字 '~' を超えていなければ、
　c + SHIFT の演算結果を main関数に返す。

　演算結果が '~' を超える場合は差分を求め、表示可能文字の最初の文字 ' ' に、
　その差分を足すことで、' ' ～ '~' の範囲で文字のシフトが可能である。

【復号化の場合】
16-23行目
- char型関数 decryption() は、char型の引数を受け取り変数 c に代入。
- if文。
　　条件文　c - SHIFT >= ' '

　変数 c から SHIFT(8) を引いた値が ASCIIコードの表示可能文字 ' ' 以下でなければ、
　c - SHIFT の演算結果を main関数に返す。

　演算結果が ' ' 以下になる場合は差分を求め、表示可能文字の最後の文字 '~' から、
　その差分を引くことで、暗号化された文の復号化が可能である。

rot13のプログラムでは、暗号化、復号化プログラムは一つのプログラムであったが、
オリジナルの暗号化、復号化プログラムはそれぞれ別のプログラムとし、SHIFTの分だけ文字を
シフトし暗号化。暗号文をシフトした分、元に戻し復号化とする。
また、シフトする文字数を #difine で、定義したので、SHIFT の値を任意に変更することができ
るようにした。

考　察

今回は、関数の定義や、islower()、toupper()などのライブラリ関数を用いて、小文字を大文字に変換したり、それを応用して暗号化・復号化プログラムを作成した。

'a' <= c && c <= 'z'などの条件文はライブラリ関数の islower()を使うことでスマートで、分かりやす条件文になり、小文字を大文字にする条件文は toupper() を用いることで、複雑な条件文と同様の処理をすることが可能になる。

また、ライブラリ関数とASCIIコード表をうまく組み合わせて使うことで、文章を暗号化したり、復号化するプログラムができた。今回作成した、オリジナルの暗号化・復号化プログラムは、文字をシフトしただけなので、しらみつぶしにシフトしていけば、何文字シフトしたか簡単に、分かってしまうので、全然実用的ではないと思う。
暗号化する際のアルゴリズムに、論理演算子や算術演算子を組み合わせることで、もっと簡単で複雑な暗号化・復号化プログラムができるのではないかと思われる。

感　想

今回のプログラミングのレポートは、１、２は比較的楽に解くことができたけど、オリジナルの暗号化・復号化プログラムを考えるのは、結構楽しかったが、時間がかかった。
いろいろアイディアは出てきたのですが、暗号化プログラムはできても、復号化プログラムが作れなかったりと非常に大変でした。

参考文献

- Steve Oualline 著、谷口功 訳　『C実践プログラミング　第3版』　オーム社
- 初心者のためのポイント学習C言語　http://www9.plala.or.jp/sgwr-t/（ 09/06/07 access ）