I117（22）パーセントエンコーディング

知念

北陸先端科学技術大学院大学 情報科学研究科
School of Information Science,
Japan Advanced Institute of Science and Technology

パーセントエンコーディング

- URL や URI で用いられるエンコード規則
  - ◇ RFC3986 で規定されている
- URL や URI には様々な区切り文字が存在
  : / ? # [ ] @
  ! $ & ' ( ) * + , ; =
- URL や URI に使えない文字(空白や漢字)も存在
- これらを扱う際のエンコード規則

※ ここでは単純化しておく、詳細は RFC3986 参照

パーセントエンコーディング *(cont.)*

該当文字を % と 16進表記にする

```
      foo△bar ⇒ foo%20bar
&i117@JAIST ⇒ %26i117%40JAIST
      北陸先端 ⇒ %CB%CC%CE%A6%C0%E8%C3%BC
```

（△は空白を示す）

ここでは漢字は EUC として処理している
JIS 等、他の文字コードでは異なる結果になるだろう

```
int percent_encode(char*dst,int dlen,char*src){
    static char hex[] = "0123456789ABCDEFZ";
    static char delim[] = ":/?#[]@!$&'()*+,;=";
    int  c=0;  int  doen;
    unsigned char *p = (unsigned char*)src,
        *q = (unsigned char*)dst, *s;
    while(*p && c<dlen-1) {
        doen = 0;
        s = (unsigned char*)delim;
        while(*s && *s!=*p) { s++; }
        if(*s==*p)   {  doen = 1;  }
        else
        if(*p>=0x80) {  doen = 1;  }
```

```
        if(doen) {
            *q++ = '%'; c++;
            if(c>=dlen-1) { goto loopend; }
            *q++ = hex[*p/16]; c++;
            if(c>=dlen-1) { goto loopend; }
            *q++ = hex[*p%16]; c++;
            if(c>=dlen-1) { goto loopend; }
            p++;
            continue;
        }
        *q++ = *p++;  c++;
    }
    *q = '\0';
#ifndef NDEBUG
```

パーセントエンコーディング *(cont.)*

```
    fprintf(stderr, "src %s\n", src);
    fprintf(stderr, "dst %s\n", dst);
#endif
}
```

漢字(EUC)を扱う可能性を考えて、関数内の文字は unsigned char とした
区切り文字かどうかの判定は while ループではなく、標準ライブラリ関数 strchr や index を使ってもよい

```
        if(strchr(delim, *p)) {  doen = 1;  }
```

```
int percent_decode(char*dst,int dlen,char*src){
    int c=0;  int hi, lo;
    unsigned char *p = (unsigned char*)src,
        *q = (unsigned char*)dst;
    while(*p && c<dlen-1) {
      if(*p==’%’) { p++;
        if(*p>=’0’&&*p<=’9’){ hi=*p-’0’;}
        else if(*p>=’A’&&*p<=’F’){ hi=*p-’A’+10;}
        else if(*p>=’a’&&*p<=’f’){ hi=*p-’a’+10;}
        else {
          fprintf(stderr, "ignore hex-letter\n");
          break;
        }
        p++;
```

```
        if(*p>='0'&&*p<='9'){ lo=*p-'0';}
        else if(*p>='A'&&*p<='F'){ lo=*p-'A'+10;}
        else if(*p>='a'&&*p<='f'){ lo=*p-'a'+10;}
        else {
            fprintf(stderr, "ignore hex-letter\n");
            break;
        }
        p++;
        *q++ = hi*16+lo;  c++;
        continue;
    }
    *q++ = *p++;
    c++;
}
```

パーセントエンコーディング *(cont.)*

```
    *q = '\0';

#ifndef NDEBUG
    fprintf(stderr, "src %s\n", src);
    fprintf(stderr, "dst %s\n", dst);
#endif
}
```

パーセントエンコーディング *(cont.)*

16進数から文字を作る箇所はマクロにもできる

```
#define H2D(x, y) \
  if((x)>='0'&&x<='9') { y = (x)-'0';} \
  else if((x)>='A'&&(x)<='F') { y = (x)-'A'+10; } \
  else if((x)>='a'&&(x)<='f') { y = (x)-'a'+10; } \
  else { fprintf(stderr, "ignore hex-letter\n"); \
    break; }
...
        H2D(*p, hi);    p++;
        H2D(*p, lo);    p++;
```

パーセントエンコーディング単体実行

```
% cat /tmp/h
北陸先端
% nkf -e /tmp/h | od -t x1
0000000 cb cc ce a6 c0 e8 c3 bc
0000010
% nkf -e /tmp/h | ./a.out
src 北陸先端
dst %CB%CC%CE%A6%C0%E8%C3%BC
src %CB%CC%CE%A6%C0%E8%C3%BC
dst 北陸先端
```

URL 合成

URL でパラメータを渡す場合はパラメータ開始を ？ で、各パラメータの名前と値を = で区切る
例えば http://jugemu/x.cgi へ以下のようなパラメータ群を渡す場合

名前	値
email	i117@jaist
name	太郎

URL は以下のようになる（単純な合成）

```
http://jugemu/x.cgi?email=i117@jaist&name=太郎
```

URL 合成 *(cont.)*

しかし、@ は区切り文字で漢字は使えない文字なので、変換が必要となる

```
http://jugemu/x.cgi?email=i117@jaist&name=太郎
                          ↓
http://jugemu/x.cgi?email=i117%40jaist&name=%C2%C0%CF%BA
```

単順に全体的を一括エンコードしては構造が分からなくなる点に注意
全体的な構成は損なわず、パラメータの値だけエンコードする
パラメータは辞書で受けとる

```
int mkurl(char *dst, int dlen,
          char *base, sdict_a *params){
  sdict_c *p;
  char tmpin[BUFSIZ], tmpout[BUFSIZ];
  int c, cw, i;
  c = 0;  cw = strlen(base);
  if(c+cw>=dlen) {
    fprintf(stderr,
      "not enough length of destination string\n");
    return -1;
  }
  strcpy(dst, base);  c = cw;
  if(!params || ! params->slot) { return 0;  }
```

パラメータがなければここで終了

```
  strcat(dst, "?");  c += 1;
  for(i=0;i<params->use;i++) {
    p = &params->slot[i];
    percent_encode(tmpin, BUFSIZ-1, p->key);
    strcpy(tmpout, tmpin);
    strcat(tmpout, "=");
    percent_encode(tmpin, BUFSIZ-1, p->value);
    strcat(tmpout, tmpin);
    cw = strlen(tmpout);
    if(i<params->use-1) {
      strcat(tmpout, "&");  cw += 1; }
```

パラメータの名前と値をエンコードして、= で連結
最終パラメータでなければ & で連結

```
    if(c+cw>=dlen) {
      fprintf(stderr,
        "not enough length of destination string\n");
      return -1;
    }
    strcat(dst, tmpout);  c += cw;
  }
}
```

テンプレートで使った readrule() を使ってパラメータを読み込む

```
sdict_a *param;
...
param = sdict_new();  sdict_init(param);
readrule(param, "rule");
mkurl(out1, BUFSIZ, "http://jugemu/x.cgi", param);
printf("out ’%s’\n", out1);
...
```

URL 合成実行結果

パラメータ内容

```
% cat rule
name=太郎
email=i117@jaist
```

コンパイルと実行

```
% cc -DNDEBUG URLen02.c percenten03.c libsdict.c readrule.c
(略)
% ./a.out
out 'http://jugemu/x.cgi?email=i117%40jaist&name=%C2%C0%CF%BA'
```

URL 合成実行結果 *(cont.)*

NDEBUG を付けずにコンパイルすると実行時に変換の様子がわかる

```
% ./a.out
src email
dst email
src i117@jaist
dst i117%40jaist
src name
dst name
src 太郎
dst %C2%C0%CF%BA
out 'http://jugemu/x.cgi?email=i117%40jaist&name=%C2%C0%CF%BA'
```

不適切な実装

パラメータの名前と値の組み全体をエンコードすると以下のようになる

```
% ./a.out
src email=i117@jaist
dst email%3Di117%40jaist
src name=太郎
dst name%3D%C2%C0%CF%BA
out 'http://anywhere.jp/foo.cgi?email%3Di117%40jaist&name%3D%C2
```

構造がなくなってパース困難

- パラメータの値に&や=が入っていたら読み間違える

演習

1) パーセントエンコーディングされている WWW ページを見付けよ★
2) それらの URL (WWWページ内部ではない) の構造を調べよ★