プログラミング言語演習 IV　6 回目　　2024 年 5 月 24 日

先週のサンプルプログラム mycat1.c では、標準入出力を使っていたため、フィルタコマンドのように使うことができるが、テキストファイルの内容を入力としたい場合にはリダイレクトなどが必要になる問題があった。mycat1 < input.txt をアーギュメントで指定して mycat1 input.txt の様に実行したい場合についてまず説明する。

mycat1.c　　標準入力の文字列をそのまま標準出力に書き出す

```
#include <stdio.h>
int main(void)
 {
 char c;                          // 1 文字ずつ読み込む時のキャラクタ
 while((c=getc(stdin))!=EOF)      // while ループで読む文字がファイルの末尾になるまで
   {                              // 標準入力から一字ずつ読む
   putc(c,stdout);                // 読んだ文字を標準出力に書き出す
   }
 return 0;
 }
```

---

mycat1.c では、標準入出力を使うため、getc(stdin)と書く代わりに getchar()としたり putc(c,stdout)と書く代わりに putchar(c)とした方が簡潔にはなる。今週は引き続き getc, putc を使い、stdin や stdout の代わりに定義するファイルハンドルというものを記述することを学ぶ。
コンパイルしたプログラムを実行する際に、アーギュメントを取る方法から説明する。gcc -o mc2 mc2.c としてコンパイルした mc2 を実行する際に、mc2 foo.txt として mc2 の読み込むファイル名(foo.txt)を実行時に指定できるようなプログラムを記述する。そのために、このプログラムのメイン関数を定義する際に、

```
int main(int argc, char **argv)
```

と記述する。整数値 argc は、コマンド名自身を含む実行時のアーギュメントの数で上記の例では 2 となる（mc2 と foo.txt）それぞれのアーギュメントの値（文字列）は argv に順に入り、上記の例では argv[0]に mc2 が、argv[1]に foo.txt が入る。3 つ以上のアーギュメントがある場合には以下同様に argv[2],argv[3]と入っていく。この辺りはシェルスクリプトの位置変数$1, $2 に類似するものとなっている。

mc2.c　　一つ目のアーギュメントをそのまま出力する

```
#include <stdio.h>
int main(int argc, char **argv)
 {
 if(argc != 2)   // コマンド名以外のアーギュメントの数が 1 でなければ
  {
  printf("Error: Wrong number of argument¥n");
  return 1;      // 異常終了
  }
 printf("Will open %s¥n",argv[1]);
 ////////////////////////////////////// ここに以下のコードを挿入
 return 0;
 }
```

これでコマンド名の次に記述するアーギュメントをプログラム内で argv[1]という文字列変数として扱えることが確認できた。次に、このアーギュメントで指定するファイル名のファイルを読み取りファイルとして開き、ファイルの内容を getc で 1 文字ずつ読み取ることができる様にする。そのために FILE *infile;としてファイルハンドルという特殊な型の変数をまず宣言し、読み取ったファイル名でファイルを開く。

```
FILE *infile;
infile = fopen(argv[1],"r");
```

これで、アーギュメントで指定したファイルから情報を読み取る準備ができた。あとは mycat1.c と同様に、stdin という標準入力のハンドルを infile というファイルハンドルで置き換えればファイルから 1 文字ずつ読み取る操作が行える。

存在しないファイル名を指定した場合など、fopen ができなかった場合には NULL を返す。その場合にはそこでエラーメッセージを出力してプログラムを停止する。

```
if(infile == NULL)
 {
 printf("Failed to open input file %s¥n",argv[1]);
 return 1;
 }
```

課題提出用のファイル名の付け方について

NumName07Job.c

Num は 3 年生は自分の学生番号の下 3 桁、過年度生は 999
Name は自分の名前、YamadaTaro などで置き換える（同姓の方がいるのでフルネーム）
Job はそれぞれの課題の名前
として

012YamadaTaro07mc2.c

の様につける。
メールの宛先は knakamura.maebit@gmail.com
メールのタイトルは日本語で自分の名前と 07 の数字を入れる　山田太郎 07

課題１．ジョブ名：mc2（必須：講義時間内）
　mc2.c と mycat1.c を参考にアーギュメントとしてファイル名をとって、mc3　foo.txt と書けば foo.txt の中身が標準出力に出るプログラム mc3.c を記述して ###name07mc2.c （name は自分の名前で置き換え）というファイル名で提出しなさい。

課題２．　ジョブ名：mywc1（必須：講義時間内）
　アーギュメントとしてファイル名をとって、mywc1　foo.txt と書けば、foo.txt の行数と文字数（改行を含む）を表示するプログラム mywc1.c を作成し ###name07mywc1.c という名前で提出しなさい。

課題３．ジョブ名：mc4（必須：講義時間内）

　アーギュメントがあればそれを入力ファイルとし、なければ標準入力から読み込んだ内容を標準出力にそのまま出すプログラム mc4.c を記述して、###ame07mc4.c というファイル名で提出しなさい。ヒント：FILE *infile;を定義した状態で、infile=stdin; とすれば、infile からの getc で標準入力からの読み込みができる。

応用課題１.ジョブ名：mc5（任意：提出期限 5 月 26 日）
　任意の数のアーギュメントをとり、たとえば mc5　foo.txt　f2.txt　f3.txt　と書けば foo.txt,　f2.txt, f3.txt の中身が連結（concatenate)されて標準出力に出るプログラム　mc5.c　を記述して　###name07mc5.c（name は自分の名前で置き換え）というファイル名で提出しなさい（アーギュメントがなければ課題２と同様)。

応用課題２.ジョブ名：mywc2（任意：提出期限 5 月 26 日）
　アーギュメントとしてファイル名をとり、　mywc2　foo.txt　と書けば foo.txt の行数と語数（スペースまたはタブまたは改行で区切られた文字列の数）と文字数の三つの整数値を出力するプログラム mywc2.c を記述し、###name07mywc2.c という名前で提出しなさい。余力があれば課題 3 と同様に標準入力に対応できる、あるいは応用課題１と同様に、複数のアーギュメントのそれぞれに対して、行数・語数・字数　ファイル名、の４カラムを Linux コマンドの wc と同様に列挙する仕様としても可。