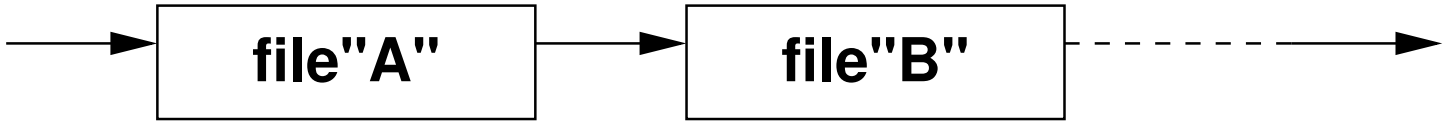
I117（9）テキストファイル（その2）

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

複数ファイル読み込み（直列）




```
fp = fopen("A", "r");
while(fgets(line, BUFSIZ, fp)) {
        ... }
fclose(fp);
fp = fopen("B", "r");
while(fgets(line, BUFSIZ, fp)) {
        ... }
fclose(fp);
...
```

ループで複数ファイルを扱う

- 重複をさけ、繰り返しをループにする
- ファイル名以外は同じ
  - ファイル名は配列に蓄える

```
for(i=0;i<filelistnum;i++) {
    fp = fopen(flist[i],"r");
    while(fgets(line, BUFSIZ, fp)) {
        ... }
    fclose(fp);
}
```

argc, argv

プログラムへ引数を与える枠組多くのプログラムで利用されている

```
int main(int argc, char *argv[]) {
...
```

- argc には引数の数
- argv には引数が入っている
  - argv[0] はプログラムの名前

ファイル名を渡すのに便利

argc, argv *(cont.)*

引数を印刷するだけのプログラム

```
#include <stdio.h>
int main(int argc, char *argv[]) {
  int i;
  for(i=0;i<argc;i++)
    printf("%d %s\n", i, argv[i]); }
```

```
% ./a.out A B
0 ./a.out
1 A
2 B
```

argc, argv *(cont.)*

合成すると... argv[0] はスキップ

```
#include <stdio.h>
int main(int argc, char *argv[]) {
  char  line[BUFSIZ];
  int   i;  FILE *fp;
  for(i=1;i<argc;i++) {
    fp = fopen(argv[i],"r");
    while(fgets(line, BUFSIZ, fp)) {
        ... }
    fclose(fp);   } }
```

複数ファイル行数計算

```
#include <stdio.h>
int main(int argc, char *argv[]) {
  char line[BUFSIZ];
  FILE *fp; int i, lc=0;
  for(i=1;i<argc;i++) {
    fp = fopen(argv[i],"r");
    while(fgets(line, BUFSIZ, fp)) {
        lc++; }
    fclose(fp);
  }
  printf("%d\n",lc);
}
```

複数ファイル行数計算 *(cont.)*

実行例、wcとの比較

```
% ./a.out xargcv.c nlc01.c
21
% wc xargcv.c nlc01.c
       8      18     138 xargcv.c
      13      30     243 nlc01.c
      21      48     381 total
```

処理を明解にするため、関数に分離する

複数ファイル行数計算 *(cont.)*

指示されたファイルの行数を数える

```
#include <stdio.h>
int linecount(char *fname) {
    char line[BUFSIZ];
    FILE *fp; int lc=0;
    fp = fopen(fname,"r");
    while(fgets(line, BUFSIZ, fp)) {
        lc++;  }
    fclose(fp);
    return lc;  }
```

複数ファイル行数計算 *(cont.)*

main は linecount を呼び出し、合計を求める

```
int main(int argc, char *argv[])
{
  int i, lc, sum=0;
  for(i=1;i<argc;i++) {
    lc = linecount(argv[i]);
    sum += lc;
  }
  printf("%d\n",sum);
}
```

複数ファイル行数計算 *(cont.)*

```
% ./a.out *.c
40
% wc *.c
      12      30     245 nlc01.c
      20      46     357 nlc01b.c
       8      18     138 xargcv.c
      40      94     740 total
```

エラーハンドリング

- fopen の失敗

```
    fp = fopen(fname,"r");
    if(fp==NULL)
        return -1;
```

- linecount の失敗

```
    lc = linecount(argv[i]);
    if(lc<0)
        continue;
```

最小限のエラー処理、メッセージを出すことも考える

エラーハンドリング *(cont.)*

シンプルなメッセージで十分

```
fp = fopen(fname,"r");
if(fp==NULL) {
  fprintf(stderr,
    "can not open file %s\n", fname);
  return -1;
}
```

```
% ./a.out ignorefilename *.c
can not open file ignorefilename
109
```

エラーハンドリング *(cont.)*

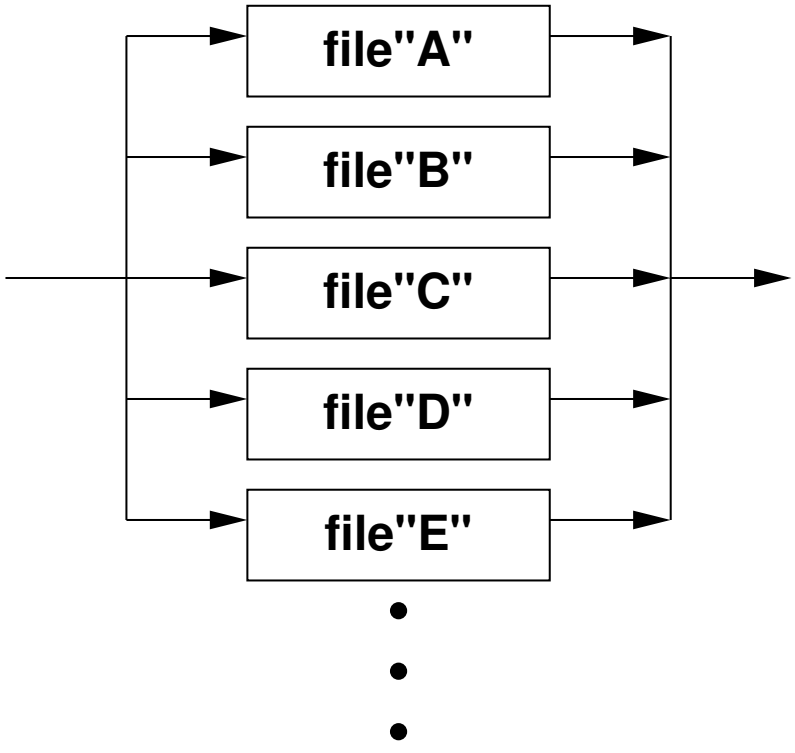
ちなみに、wc も似たようもの

```
% wc ignorefilename *.c
wc: ignorefilename: No such file or direct
      12       30      245 nlc01.c
      20       46      357 nlc01b.c
      34       55      455 nlc01c.c
      35       62      507 nlc01d.c
       8       18      138 xargcv.c
     109      211     1702 total
```

並列複数ファイル読み込み

同時に複数ファイルを扱う
ファイル数分情報を確保する

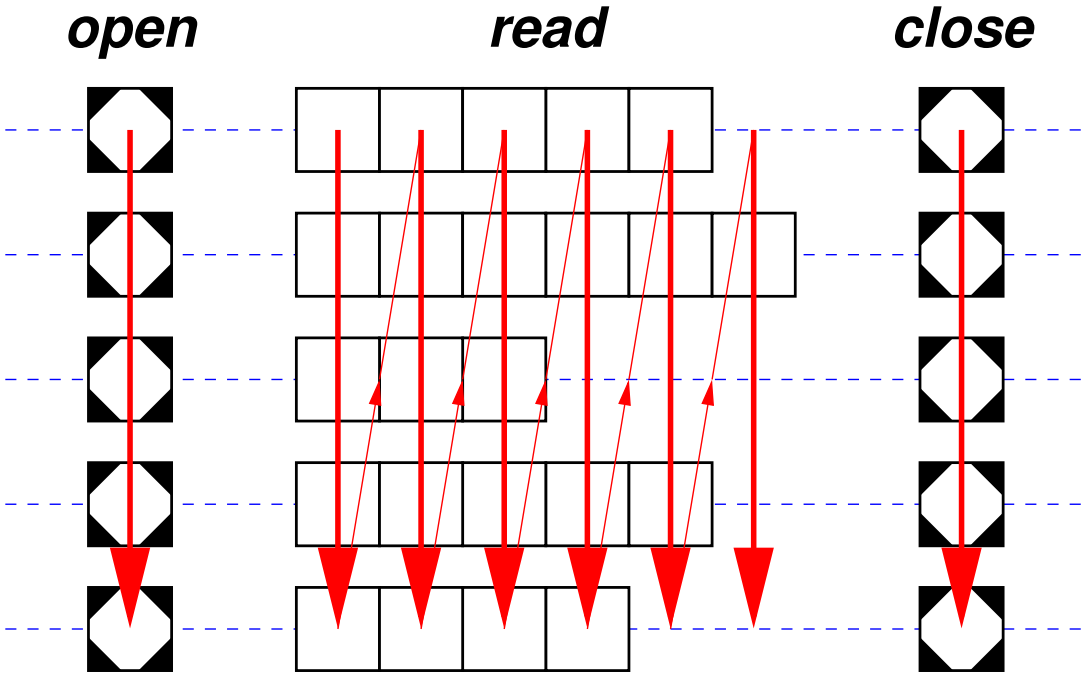
```
typedef struct {
    char *fname;
    FILE *fp;
    char  line[BUFSIZ];
    int   lc;
    int   reach_eof;
} lcparam;
```



reach_eof は最後まで読んだかどうかを記録する

並列複数ファイル読み込み *(cont.)*

全ファイルを読み終るまで繰り返す



ファイル長がバラバラなので、最後に達したかどうか判定が必要

並列複数ファイル読み込み *(cont.)*

```
lcparam *lclist;
lclist = (lcparam*)malloc(
    sizeof(lcparam)*(argc));
```

```
for(i=1;i<argc;i++) {
    lc = openfile(&lclist[i], argv[i]);
    if(lc<0) {
        continue;
    }
}
```

```
do {
    /* process reading */
    for(i=1;i<argc;i++) {
        lc = linecount(&lclist[i]);
    }
    /* check all file was read */
    e = 0;
    for(i=1;i<argc;i++) {
        if(lclist[i].reach_eof>0) {
            e++;
        }
    }
} while(e<argc-1);
```

並列複数ファイル読み込み *(cont.)*

```
sum = 0;
for(i=1;i<argc;i++) {
  printf("%s %d\n",
    lclist[i].fname, lclist[i].lc);
  sum += lclist[i].lc;
  if(lclist[i].fp) {
    fclose(lclist[i].fp);
  }
}
printf("total %d\n",sum);
```

並列複数ファイル読み込み *(cont.)*

```
int openfile(lcparam *lcp,
  char *fname) {
  FILE *fp;
  lcp->reach_eof = -1;
  lcp->fp = NULL; lcp->lc = 0;
  lcp->fname = strdup(fname);
  fp = fopen(fname,"r");
  if(fp==NULL) {
    fprintf(stderr,
      "can not open file %s\n", fname);
```

並列複数ファイル読み込み *(cont.)*

```
    lcp->reach_eof = 1;
    return -1;
  }
  lcp->fp = fp;
  lcp->reach_eof = 0;
  return 0;
}
```

失敗した場合には -1、成功したら 0 を返す

並列複数ファイル読み込み *(cont.)*

```
int linecount(lcparam *lcp) {
  if(lcp->reach_eof!=0) {
      return -1;
  }
  if(fgets(lcp->line,BUFSIZ,lcp->fp)){
    lcp->lc++;
    return 0;
  }
  lcp->reach_eof = 1;
  return 1;  }
```

並列複数ファイル読み込み *(cont.)*

結果

```
% ./a.out *.c
nlc01.c 12
nlc01b.c 20
nlc01c.c 34
nlc01d.c 35
plc01.c 90
plc01b.c 87
xargcv.c 8
total 286
```

演習

1) 直列複数ファイル行数計上のプログラムで各ファイルの行数を表示するよう改造せよ★
   - (BUFSIZより)長い行は考慮しなくて良い
2) 同様に各ファイルのバイト数を表示するよう改造せよ★
3) 同様に各ファイルの単語数を表示するよう改造せよ★★
   - プログラム wc と同様の出力にせよ

演習 *(cont.)*

4) 並列複数ファイル読み込みの別実装を作れ★
個々のファイルを読み終るとすぐに close を実施する