Day21.C

File Processing

Young W. Lim

December 12, 2017

1 Sequential and Random Access of Files

1.1 Sequential Access of Files

```
::::::::::::::
h1.c
::::::::::::::
#include <stdio.h>

int main(void) {

  FILE *fp;
  int   A[26];
  int   i;

  fp = fopen("test.dat", "wb");

  for (i=0; i<26; ++i) {
    A[i] = i + 0x41;
    A[i] |= (A[i] << 8);
    A[i] |= (A[i] << 8);
    A[i] |= (A[i] << 8);

    printf("%#x\n", A[i]);
  }

  fwrite(A, sizeof(int), 26, fp);


  for (i=0; i<26; ++i) {
    A[i] += 0x20202020;

    printf("%#x\n", A[i]);

  }

  fwrite(A, sizeof(int), 26, fp);

  fclose(fp);

}

::::::::::::::
out
::::::::::::::
0x41414141
0x42424242
0x43434343
0x44444444
0x45454545
```

```
0x46464646
0x47474747
0x48484848
0x49494949
0x4a4a4a4a
0x4b4b4b4b
0x4c4c4c4c
0x4d4d4d4d
0x4e4e4e4e
0x4f4f4f4f
0x50505050
0x51515151
0x52525252
0x53535353
0x54545454
0x55555555
0x56565656
0x57575757
0x58585858
0x59595959
0x5a5a5a5a
0x61616161
0x62626262
0x63636363
0x64646464
0x65656565
0x66666666
0x67676767
0x68686868
0x69696969
0x6a6a6a6a
0x6b6b6b6b
0x6c6c6c6c
0x6d6d6d6d
0x6e6e6e6e
0x6f6f6f6f
0x70707070
0x71717171
0x72727272
0x73737373
0x74747474
0x75757575
0x76767676
0x77777777
0x78787878
0x79797979
0x7a7a7a7a
::::::::::::::
xxd test.dat
::::::::::::::
```

```
00000000: 4141 4141 4242 4242 4343 4343 4444 4444  AAAABBBBCCCCDDDD
00000010: 4545 4545 4646 4646 4747 4747 4848 4848  EEEEFFFFGGGGHHHH
00000020: 4949 4949 4a4a 4a4a 4b4b 4b4b 4c4c 4c4c  IIIIJJJJKKKKLLLL
00000030: 4d4d 4d4d 4e4e 4e4e 4f4f 4f4f 5050 5050  MMMMNNNNOOOOPPPP
00000040: 5151 5151 5252 5252 5353 5353 5454 5454  QQQQRRRRSSSSTTTT
00000050: 5555 5555 5656 5656 5757 5757 5858 5858  UUUUVVVVWWWWXXXX
00000060: 5959 5959 5a5a 5a5a 6161 6161 6262 6262  YYYYZZZZaaaabbbb
00000070: 6363 6363 6464 6464 6565 6565 6666 6666  ccccddddeeeeffff
00000080: 6767 6767 6868 6868 6969 6969 6a6a 6a6a  ggggghhhhiiiijjjj
00000090: 6b6b 6b6b 6c6c 6c6c 6d6d 6d6d 6e6e 6e6e  kkkkllllmmmmnnnn
000000a0: 6f6f 6f6f 7070 7070 7171 7171 7272 7272  ooooppppqqqqrrrr
000000b0: 7373 7373 7474 7474 7575 7575 7676 7676  ssssttttuuuuvvvv
000000c0: 7777 7777 7878 7878 7979 7979 7a7a 7a7a  wwwwxxxxyyyyzzzz
```

Writing Sequentially

- a character has a size of 1-byte
- an integer has a size of 4-byte
- `i+0x41 (i=0, ..., 25)` can enumerate all the capital letters (A to Z)
- an integer array element `A[i]` contains 4 copies of `i+41`
- to enumerate all the lower case letters, add `0x20202020` to `A[i]`

```
::::::::::::::
h2.c
::::::::::::::
#include <stdio.h>

int main(void) {

  FILE *fp;
  int   A[26];
  int   i;

  fp = fopen("test.dat", "rb");

  fread(A, sizeof(int), 26, fp);

  for (i=0; i<26; ++i) {
    printf("%#x\n", A[i]);
  }

  fread(A, sizeof(int), 26, fp);

  for (i=0; i<26; ++i) {
    printf("%#x\n", A[i]);
  }

  fclose(fp);
```

```
}

::::::::::::::
xxd test.dat
::::::::::::::
00000000: 4141 4141 4242 4242 4343 4343 4444 4444  AAAABBBBCCCCDDDD
00000010: 4545 4545 4646 4646 4747 4747 4848 4848  EEEEFFFFGGGGHHHH
00000020: 4949 4949 4a4a 4a4a 4b4b 4b4b 4c4c 4c4c  IIIIJJJJKKKKLLLL
00000030: 4d4d 4d4d 4e4e 4e4e 4f4f 4f4f 5050 5050  MMMMNNNNOOOOPPPP
00000040: 5151 5151 5252 5252 5353 5353 5454 5454  QQQQRRRRSSSSTTTT
00000050: 5555 5555 5656 5656 5757 5757 5858 5858  UUUUVVVVWWWWXXXX
00000060: 5959 5959 5a5a 5a5a 6161 6161 6262 6262  YYYYZZZZaaaabbbb
00000070: 6363 6363 6464 6464 6565 6565 6666 6666  ccccddddeeeeffff
00000080: 6767 6767 6868 6868 6969 6969 6a6a 6a6a  ggggghhhhiiiijjjj
00000090: 6b6b 6b6b 6c6c 6c6c 6d6d 6d6d 6e6e 6e6e  kkkkllllmmmmnnnn
000000a0: 6f6f 6f6f 7070 7070 7171 7171 7272 7272  ooooppppqqqqrrrr
000000b0: 7373 7373 7474 7474 7575 7575 7676 7676  ssssttttuuuuvvvv
000000c0: 7777 7777 7878 7878 7979 7979 7a7a 7a7a  wwwwxxxxyyyyzzzz
::::::::::::::
out
::::::::::::::
0x41414141
0x42424242
0x43434343
0x44444444
0x45454545
0x46464646
0x47474747
0x48484848
0x49494949
0x4a4a4a4a
0x4b4b4b4b
0x4c4c4c4c
0x4d4d4d4d
0x4e4e4e4e
0x4f4f4f4f
0x50505050
0x51515151
0x52525252
0x53535353
0x54545454
0x55555555
0x56565656
0x57575757
0x58585858
0x59595959
0x5a5a5a5a
0x61616161
0x62626262
```

```
0x63636363
0x64646464
0x65656565
0x66666666
0x67676767
0x68686868
0x69696969
0x6a6a6a6a
0x6b6b6b6b
0x6c6c6c6c
0x6d6d6d6d
0x6e6e6e6e
0x6f6f6f6f
0x70707070
0x71717171
0x72727272
0x73737373
0x74747474
0x75757575
0x76767676
0x77777777
0x78787878
0x79797979
0x7a7a7a7a
```

Reading Sequentially

- after reading integers consisting of 26 capital letters,
  read integers consisting 26 small case letters.
- the array `A` has a size of 26.
- the previous data will be overwritten by the second read operations

1.2 Random Access of Files

```
::::::::::::::
h3.c
::::::::::::::
#include <stdio.h>

int main(void) {

  FILE *fp;
  int   A[26];
  int   i;

  for (i=0; i<26; ++i) {
    A[i] = i + 0x41;
    A[i] |= (A[i] << 8);
    A[i] |= (A[i] << 8);
```

```
    A[i] |= (A[i] << 8);
  }

  fp = fopen("test.dat", "wb");


  // ..................................
  // SEEK_SET : offset from the beginning
  // SEEK_CUR : offset from the current
  // SEEK_END : offset from the end
  // ..................................



  for (i= 0; i<26; ++i) {
    if (i%2 == 0) {
      fseek(fp, i*4, SEEK_SET);
      fwrite(&A[i], sizeof(int), 1, fp);
      printf("%#x\n", A[i]);
    }
  }

  fclose(fp);
}
::::::::::::::
out
::::::::::::::
0x41414141
0x43434343
0x45454545
0x47474747
0x49494949
0x4b4b4b4b
0x4d4d4d4d
0x4f4f4f4f
0x51515151
0x53535353
0x55555555
0x57575757
0x59595959
::::::::::::::
xxd test.dat
::::::::::::::
00000000: 4141 4141 0000 0000 4343 4343 0000 0000  AAAA....CCCC....
00000010: 4545 4545 0000 0000 4747 4747 0000 0000  EEEE....GGGG....
00000020: 4949 4949 0000 0000 4b4b 4b4b 0000 0000  IIII....KKKK....
00000030: 4d4d 4d4d 0000 0000 4f4f 4f4f 0000 0000  MMMM....OOOO....
00000040: 5151 5151 0000 0000 5353 5353 0000 0000  QQQQ....SSSS....
00000050: 5555 5555 0000 0000 5757 5757 0000 0000  UUUU....WWWW....
00000060: 5959 5959                                YYYY
```

```
::::::::::::::
h4.c
::::::::::::::
#include <stdio.h>

int main(void) {

  FILE *fp;
  int   A[26];
  int   i;


  fp = fopen("test.dat", "rb");

  // ..................................
  // SEEK_SET : offset from the beginning
  // SEEK_CUR : offset from the current
  // SEEK_END : offset from the end
  // ..................................



  for (i=0; i<26; ++i) {
    if (i%2 == 0) {
      fseek(fp, i*4, SEEK_SET);
      fread(&A[i], sizeof(int), 1, fp);
      printf("0x%x\n", A[i]);
    }
  }

  fclose(fp);

}
::::::::::::::
test.out
::::::::::::::
00000000: 4141 4141 0000 0000 4343 4343 0000 0000  AAAA....CCCC....
00000010: 4545 4545 0000 0000 4747 4747 0000 0000  EEEE....GGGG....
00000020: 4949 4949 0000 0000 4b4b 4b4b 0000 0000  IIII....KKKK....
00000030: 4d4d 4d4d 0000 0000 4f4f 4f4f 0000 0000  MMMM....OOOO....
00000040: 5151 5151 0000 0000 5353 5353 0000 0000  QQQQ....SSSS....
00000050: 5555 5555 0000 0000 5757 5757 0000 0000  UUUU....WWWW....
00000060: 5959 5959                                YYYY
::::::::::::::
out
::::::::::::::
0x41414141
0x43434343
0x45454545
0x47474747
```

```
0x49494949
0x4b4b4b4b
0x4d4d4d4d
0x4f4f4f4f
0x51515151
0x53535353
0x55555555
0x57575757
0x59595959
```