

운영체제

Reader & Writer Problem

학번 : 20143104

이름 : 조승현

```

void reset(char *fileVar) {
// fileVar라는 이름의 파일이 없을 때만
// fileVar라는 이름의 텍스트 파일을 새로 만들고 0값을 기록한다.
FILE *f = fopen(fileVar, "r");
if(f==NULL){
    f = fopen(fileVar , "w");
    fprintf(f, "%d\n", 0);
}
fclose(f);
}

void Store(char *fileVar,int i) {
// fileVar 파일 끝에 i 값을 append한다.
FILE *f = fopen(fileVar, "a+");
while(!feof(f));
fprintf(f, "%d\n", 0);
fclose(f);
}

int Load(char *fileVar) {
// fileVar 파일의 마지막 값을 읽어 온다.
int data;
FILE *f = fopen(fileVar, "r");
while(!feof(f))
    fscanf(f, "%d", &data);
fclose(f);
return data;
}

void add(char *fileVar,int i) {
// fileVar 파일의 마지막 값을 읽어서 i를 더한 후에 이를 끝에 append 한다.
int data;
pid_t pid = getpid();
FILE *f = fopen(fileVar, "a+");
while(!feof(f))
    fscanf(f, "%d", &data);
fprintf(f, "%d\n", (i+data));
fclose(f);
}

void sub(char *fileVar,int i) {
// fileVar 파일의 마지막 값을 읽어서 i를 뺀 후에 이를 끝에 append 한다.
pid_t pid = getpid();
int data;
FILE *f = fopen(fileVar, "a+");
while(!feof(f))
    fscanf(f, "%d", &data);
fprintf(f, "%d\n", (data-i));
fclose(f);
}

```

```

// Class CondVar
typedef struct _cond {
    int semid;
    char *queueLength;
} CondVar;
void initCondVar(CondVar *c, key_t semkey, char *queueLength) {
    c->queueLength = queueLength;
    reset(c->queueLength); // queueLength=0
    if ((c->semid = initsem(semkey,0)) < 0)
        // 세마포를 연결한다.(없으면 초기값을 0로 주면서 새로 만들어서 연결한다.)
        exit(1);
}
void Wait(CondVar *c, Lock *lock) {
    //P연산을 통해 CondVar의 wait 큐에 추가한다.
    add(c->queueLength, 1);
    Release(lock);
    p(c->semid);
    Acquire(lock);
}
void Signal(CondVar *c) {
    //V연산을 통해 CondVar의 큐에 하나만을 깨운다.
    if(Load(c->queueLength) >0){
        v(c->semid);
        sub(c->queueLength, 1);
    }
}
void Broadcast(CondVar *c) {
    //V연산을 통해 CondVar의 큐에 모든 것을 깨운다.
    while(Load(c->queueLength) > 0){
        v(c->semid);
        sub(c->queueLength,1);
    }
}

```

<reader.c>

```
void main(int argc, char *argv[]){
    struct timeval bgn,end;    //시간을 기록하기 위한 변수
    gettimeofday(&bgn, NULL); //프로그램 시작 시간 기록
    int diff;
    key_t semkey = 0x200; //semaphore ID(lock)
    key_t semkey2 = 0x100; //semaphore ID(oktoread)
    key_t semkey3 = 0x300; //semaphore ID(oktowrite)
    pid_t pid;
    Lock lock;
    CondVar okToRead;
    CondVar okToWrite;
    pid = getpid();
    initLock(&lock, semkey);
    initCondVar(&okToRead, semkey2, "oktoread"); // CondVar 초기화
    initCondVar(&okToWrite, semkey3, "oktowrite");
    reset("AR");
    reset("AW");
    reset("WR");
    reset("WW");
    sleep(atoi(argv[1])); // 지연시간
    Acquire(&lock);
    while((Load("AW") + Load("WW")) > 0){ //실행중이거나 기다리는 writer 가 있으면
        gettimeofday(&end, NULL);
        diff = end.tv_sec - bgn.tv_sec;
        printf("process %d in reader waiting %ds\n", pid, diff); // waiting 시간 측정
        add("WR", 1); // reader waiting
        Wait(&okToRead, &lock); // reader waiting
        gettimeofday(&end, NULL);
        diff = end.tv_sec - bgn.tv_sec;
        printf("process %d in reader wake up %ds\n", pid, diff); // wake up 시간측정
        sub("WR",1); // reader wake up
    }
    gettimeofday(&end, NULL);
    diff = end.tv_sec - bgn.tv_sec;
    printf("process %d in reader %ds\n", pid, diff); // 실행되는 reader 시간 측정
    add("AR",1); // reader active
    Release(&lock);
    sleep(atoi(argv[2])); // 실행
    Acquire(&lock);
    gettimeofday(&end, NULL);
    diff = end.tv_sec - bgn.tv_sec;
    printf("process %d in reader exiting %ds\n", pid, diff); //Active 한reader 가 끝나는 시간측정
    sub("AR", 1); // reader closed
    if(Load("AR")==0 && Load("WW") > 0){ // 마지막 reader 의 경우 waiting writer가 있으면 writer wake up
        Signal(&okToWrite);
    }
    Release(&lock);
    exit(0);
}
```

<writer.c>

```
void main(int argc, char *argv[]){
    struct timeval bgn,end; //시간을 기록하기 위한 변수
    gettimeofday(&bgn, NULL); //프로그램 시작 시간 기록
    int diff;
    key_t semkey = 0x200; //semaphore ID(lock)
    key_t semkey2 = 0x100; //semaphore ID(oktoread)
    key_t semkey3 = 0x300; //semaphore ID(oktowrite)
    int semid;
    pid_t pid;
    Lock lock;
    CondVar okToRead;
    CondVar okToWrite;
    pid = getpid();
    initLock(&lock, semkey);
    initCondVar(&okToRead, semkey2, "oktoread"); //CondVar 초기화
    initCondVar(&okToWrite, semkey3, "oktowrite");
    reset("AR");
    reset("AW");
    reset("WR");
    reset("WW");
    sleep(atoi(argv[1])); // 지연시간
    Acquire(&lock);
    while((Load("AW") + Load("AR")) > 0){ //기다리는 writer 거나 실행중인 reader가 있으면
        gettimeofday(&end, NULL);
        diff = end.tv_sec - bgn.tv_sec;
        printf("process %d in writer waiting %ds\n", pid, diff); //waiting 시간 측정
        add("WW", 1); //reader waiting
        Wait(&okToWrite, &lock); //reader waiting
        gettimeofday(&end, NULL);
        diff = end.tv_sec - bgn.tv_sec;
        printf("process %d in writer wake up %ds\n", pid, diff); //wake up 시간 측정
        sub("WW",1); //writer wake up
    }
    gettimeofday(&end, NULL);
    diff = end.tv_sec - bgn.tv_sec;
    printf("process %d in writer %ds\n", pid, diff); //실행되는 writer 시간 측정
    add("AW", 1); //writer active
    Release(&lock);
    sleep(atoi(argv[2])); //실행
    Acquire(&lock);
    gettimeofday(&end, NULL);
    diff = end.tv_sec - bgn.tv_sec;
    printf("process %d in writer exiting %ds\n", pid, diff); //Active한 writer 끝나는 시간측정
    sub("AW", 1); //writer closed
    if(Load("WW") > 0){ //기다리는 writer가 있으면 wake up
        Signal(&okToWrite);
    }
    else if(Load("WR") > 0){ //기다리는 writer가 없고 기다리는 reader가 있으면 reader 전부 깨우기
        Broadcast(&okToRead);
    }
    Release(&lock);
    exit(0);
}
```

	1초	2초	3초	4초	5초	6초	7초	10초	13초	16초	18초
시작	R1	R2	W1	R3	R4	W2					
AR	R1(1)	R2(2)				R1(1)	R2(0)		R3(1) R4(2)	R4(1)	R3(0)
WR				R3(1)	R4(2)						
AW							W1(1)	W1(0) W2(1)	W2(0)		
WW			W1(1)			W2(2)	W1(1)	W2(0)			

<실제로 실행되는 과정을 나타낸 표>

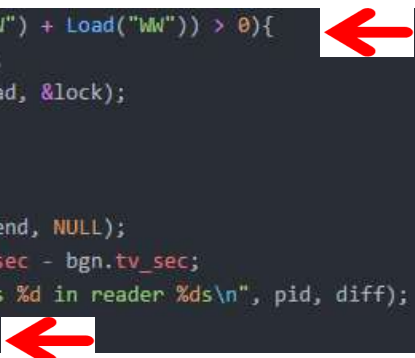
<분석>

1초 - AR=0 WR=0 AW=0 WW=0 (Reader1 실행 전)

AR=1 WR=0 AW=0 WW=0 (Reader1 실행 후)

```
while((Load("AW") + Load("WW")) > 0){
    add("WR", 1);
    Wait(&okToRead, &lock);
    sub("WR",1);
}

gettimeofday(&end, NULL);
diff = end.tv_sec - bgn.tv_sec;
printf("process %d in reader %ds\n", pid, diff);
add("AR",1);
```

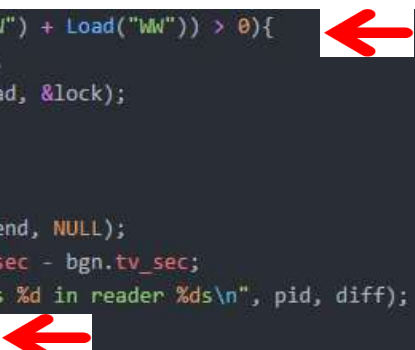


2초 - AR=1 WR=0 AW=0 WW=0 (Reader2 실행 전)

AR=2 WR=0 AW=0 WW=0 (Reader2 실행 후)

```
while((Load("AW") + Load("WW")) > 0){
    add("WR", 1);
    Wait(&okToRead, &lock);
    sub("WR",1);
}

gettimeofday(&end, NULL);
diff = end.tv_sec - bgn.tv_sec;
printf("process %d in reader %ds\n", pid, diff);
add("AR",1);
```



3초 - AR=2 WR=0 AW=0 WW=0 (Writer1 실행 전)

AR=2 WR=1 AW=0 WW=1 (Writer1 실행 후)

```
while((Load("AW") + Load("AR")) > 0){  
    add("WW", 1);  
    Wait(&okToWrite, &lock);  
    sub("WW",1);  
}  
  
gettimeofday(&end, NULL);  
diff = end.tv_sec - bgn.tv_sec;  
printf("process %d in writer %ds\n", pid, diff);  
add("AW", 1);
```

Queue	
Reader	Writer
	W1

4초 - AR=2 WR=0 AW=0 WW=1 (Reader3 실행 전)

AR=2 WR=1 AW=0 WW=1 (Reader3 실행 후)

```
while((Load("AW") + Load("WW")) > 0){  
    add("WR", 1);  
    Wait(&okToRead, &lock);  
    sub("WR",1);  
}  
  
gettimeofday(&end, NULL);  
diff = end.tv_sec - bgn.tv_sec;  
printf("process %d in reader %ds\n", pid, diff);  
add("AR",1);
```

Queue	
Reader	Writer
R3	W1

5초 - AR=2 WR=1 AW=0 WW=1 (Reader4 실행 전)

AR=2 WR=2 AW=0 WW=1 (Reader4 실행 후)

```
while((Load("AW") + Load("WW")) > 0){  
    add("WR", 1);  
    Wait(&okToRead, &lock);  
    sub("WR",1);  
}  
  
gettimeofday(&end, NULL);  
diff = end.tv_sec - bgn.tv_sec;  
printf("process %d in reader %ds\n", pid, diff);  
add("AR",1);
```

Queue	
Reader	Writer
R3	W1
R4	

6초 - AR=2 WR=2 AW=0 WW=1 (Reader1 종료 전, Writer2 실행 전)
 AR=1 WR=2 AW=0 WW=1 (Reader1 종료 후)
 AR=1 WR=2 AW=0 WW=2 (Writer2 실행 후)

```
sub("AR", 1);
if(Load("AR")==0 && Load("WW") > 0){
  Signal(&okToWrite);
}
```

```
while((Load("AW") + Load("AR")) > 0){
  add("WW", 1);
  Wait(&okToWrite, &lock);
  sub("WW", 1);
}

gettimeofday(&end, NULL);
diff = end.tv_sec - bgn.tv_sec;
printf("process %d in writer %ds\n", pid, diff);
add("AW", 1);
```

Queue	
Reader	Writer
R3	W1
R4	W2

7초 - AR=1 WR=2 AW=0 WW=2 (Reader2 종료 전)
 AR=0 WR=2 AW=1 WW=1 (Reader2 종료 후) Writer1 wake up

```
sub("AR", 1);
if(Load("AR")==0 && Load("WW") > 0){
  Signal(&okToWrite);
}
```

Queue	
Reader	Writer
R3	W2
R4	

10초 - AR=0 WR=2 AW=1 WW=1 (Writer1 종료 전)
 AR=0 WR=2 AW=1 WW=0 (Writer1 종료 후) Writer2 wake up

```
sub("AW", 1);
if(Load("WW") > 0){
  Signal(&okToWrite);
}
else if(Load("WR") > 0){
  Broadcast(&okToRead);
}
```

Queue	
Reader	Writer
R3	
R4	

13초 - AR=0 WR=2 AW=1 WW=0 (Writer2 종료 전)

AR=2 WR=0 AW=0 WW=0 (Writer2 종료 후) Reader broadcast

```
sub("AW", 1);  
if(Load("WW") > 0){  
    Signal(&okToWrite);  
}  
else if(Load("WR") > 0){  
    Broadcast(&okToRead);  
}
```

Queue	
Reader	Writer

16초 - AR=1 WR=0 AW=0 WW=0 (Reader4 종료)

```
sub("AR", 1);  
if(Load("AR")==0 && Load("WW") > 0){  
    Signal(&okToWrite);  
}
```

18초 - AR=0 WR=0 AW=0 WW=0 (Reader3 종료)

```
sub("AR", 1);  
if(Load("AR")==0 && Load("WW") > 0){  
    Signal(&okToWrite);  
}
```

<결과 1>

```
cs4096@cs4096-ThinkPad-T440:~/OS/hw2$ process 21826 in reader 1s
process 21827 in reader 2s
process 21828 in writer waiting 3s
process 21829 in reader waiting 4s
process 21830 in reader waiting 5s
process 21826 in reader exiting 6s
process 21831 in writer waiting 6s
process 21827 in reader exiting 7s
process 21828 in writer wake up 7s
process 21828 in writer 7s
process 21828 in writer exiting 10s
process 21831 in writer wake up 10s
process 21831 in writer 10s
process 21831 in writer exiting 13s
process 21829 in reader wake up 13s
process 21829 in reader 13s
process 21830 in reader wake up 13s
process 21830 in reader 13s
process 21830 in reader exiting 16s
process 21829 in reader exiting 18s
```

<결과 2>

AR

0
1
2
1
0
1
2
1
0

<AR>

AW

0
1
0
1
0

<AW>

WR

0
1
2
1
0

<WR>

WW

0
1
2
1
0

<WW>

oktoread

0
1
2
1
0

<oktoread>

oktowrite

0
1
2
1
0

<oktowrite>