**1. pthread의 mutex만을 이용한 생산자/소비자**

#include <stdio.h>

#include <pthread.h>

int buffer[100];

int count = 0;

int in = -1, out = -1;

pthread\_mutex\_t mutex = PTHREAD\_MUTEX\_INITIALIZER;

void\* producer(void \*pData);

void\* consumer(void \*pData);

int main(void)

{

int i;

pthread\_t threads[2];

pthread\_create(&threads[0], NULL, producer, NULL);

pthread\_create(&threads[1], NULL, consumer, NULL);

for (i=0; i<2; i++)

pthread\_join(threads[i], NULL);

return 0;

}

void\* producer(void \*pData)

{

int i;

for (i=0; i<300; i++)

{

while(count == 100);

pthread\_mutex\_lock(&mutex);

in++;

in %= 100;

buffer[in] = i;

count++;

pthread\_mutex\_unlock(&mutex);

printf("Produce data = %d\n", buffer[in]);

}

}

void\* consumer(void \*pData)

{

int i,data;

for (i=0; i<300; i++)

{

while(count == 0);

pthread\_mutex\_lock(&mutex);

out++;

out %= 100;

data = buffer[out];

count--;

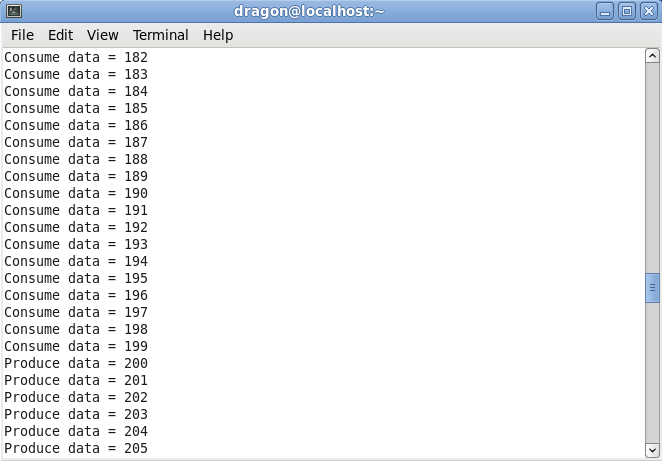
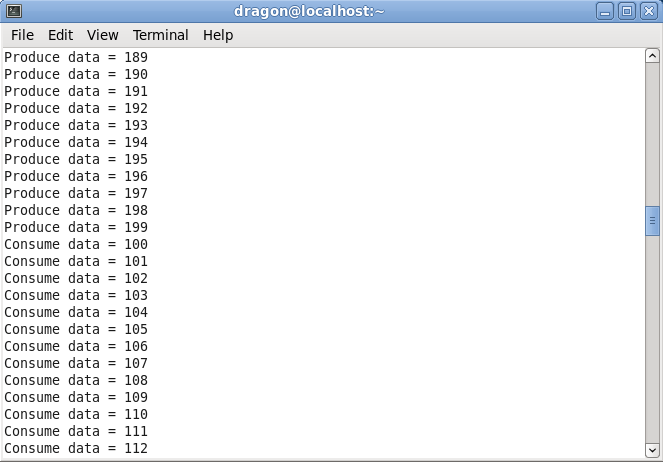
pthread\_mutex\_unlock(&mutex);

printf("Consume data = %d\n", data);

}

}

**실행결과:**



-Produce가 생산한 만큼 Consumer가 소비한다. Producer는 100개를 초과하여 생산하지 못한다.

**2. pthread의 mutex와 condition을 이용한 생산자/소비자**

#include <stdio.h>

#include <pthread.h>

int buffer[100];

int count = 0;

int in = -1, out = -1;

pthread\_mutex\_t mutex = PTHREAD\_MUTEX\_INITIALIZER;

pthread\_cond\_t buffer\_has\_space = PTHREAD\_MUTEX\_INITIALIZER;

pthread\_cond\_t buffer\_has\_data = PTHREAD\_MUTEX\_INITIALIZER;

void\* producer(void \*pData);

void\* consumer(void \*pData);

int main(void)

{

int i;

pthread\_t threads[2];

pthread\_create(&threads[0], NULL, producer, NULL);

pthread\_create(&threads[1], NULL, consumer, NULL);

for (i=0; i<2; i++)

pthread\_join(threads[i], NULL);

return 0;

}

void\* producer(void \*pData)

{

int i;

for (i=0; i<300; i++)

{

pthread\_mutex\_lock(&mutex);

if(count == 100)

pthread\_cond\_wait(&buffer\_has\_space, &mutex);

in++;

in %= 100;

buffer[in] = i;

count++;

pthread\_cond\_signal(&buffer\_has\_data);

pthread\_mutex\_unlock(&mutex);

printf("Produce data = %d\n", buffer[in]);

}

}

void\* consumer(void \*pData)

{

int i,data;

for (i=0; i<300; i++)

{

pthread\_mutex\_lock(&mutex);

if(count == 0)

pthread\_cond\_wait(&buffer\_has\_data, &mutex);

out++;

out %= 100;

data = buffer[out];

count--;

pthread\_cond\_signal(&buffer\_has\_space);

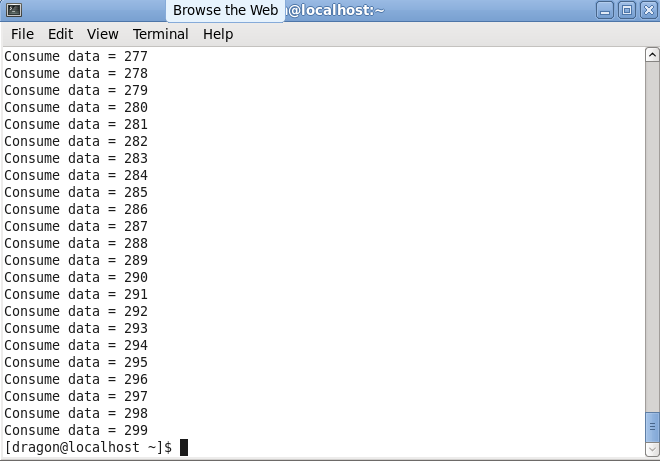
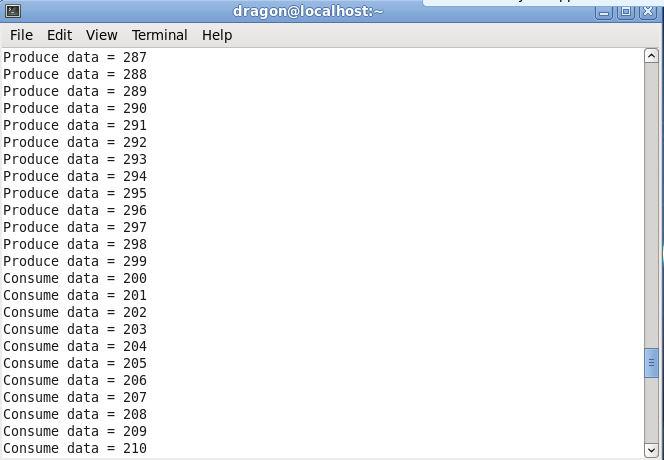
pthread\_mutex\_unlock(&mutex);

printf("Consume data = %d\n", data);

}

}

**실행결과:**



-Produce가 생산한 만큼 Consumer가 소비한다. Producer는 100개를 초과하여 생산하지 못한다.

**3. pthread의 mutex와 semaphore를 이용한 생산자/소비자**

#include <stdio.h>

#include <pthread.h>

#include <semaphore.h>

int buffer[100];

int count = 0;

int in = -1, out = -1;

pthread\_mutex\_t mutex = PTHREAD\_MUTEX\_INITIALIZER;

void\* producer(void \*pData);

void\* consumer(void \*pData);

sem\_t full, empty;

int main(void)

{

int i;

sem\_init(&empty,0,0);

sem\_init(&full,0,100);

pthread\_t threads[2];

pthread\_create(&threads[0], NULL, producer, NULL);

pthread\_create(&threads[1], NULL, consumer, NULL);

for (i=0; i<2; i++)

pthread\_join(threads[i], NULL);

return 0;

}

void\* producer(void \*pData)

{

int i;

for (i=0; i<300; i++)

{

sem\_wait(&full);

pthread\_mutex\_lock(&mutex);

in++;

in %= 100;

buffer[in] = i;

count++;

printf("Produce data = %d\n", buffer[in]);

pthread\_mutex\_unlock(&mutex);

sem\_post(&empty);

}

}

void\* consumer(void \*pData)

{

int i,data;

for (i=0; i<300; i++)

{

sem\_wait(&empty);

pthread\_mutex\_lock(&mutex);

out++;

out %= 100;

data = buffer[out];

count--;

printf("Consume data = %d\n", data);

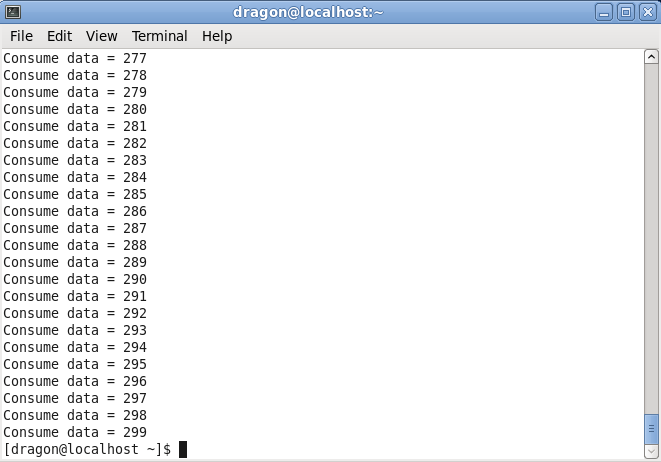
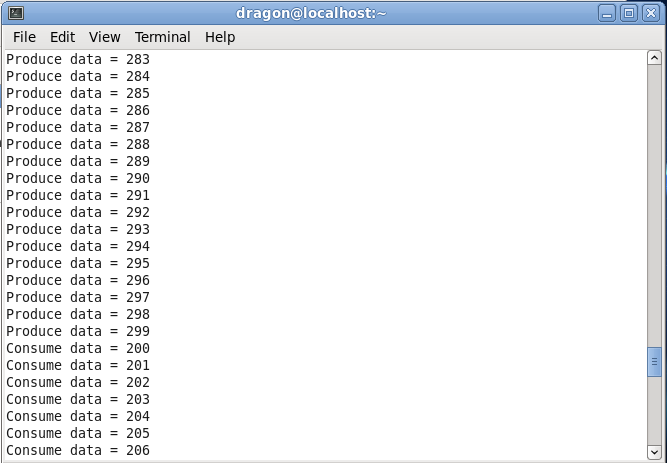
pthread\_mutex\_unlock(&mutex);

sem\_post(&full);

}

}

**실행결과 :**



-Produce가 생산한 만큼 Consumer가 소비한다. Producer는 100개를 초과하여 생산하지 못한다.