

Name: Phạm Mai Dung

ID: 19520477

Class: IT007.L21.1

## OPERATING SYSTEM LAB 5'S REPORT

### SUMMARY

Task		Status	Page
Section 1.5	Ex 1	Done	2 – 5
	Ex 2	Done	6 – 11
	Ex 3	Done	12 – 13
	Task name 4	Done	14 – 15

**Self-scores: 8**

*\*Note: Export file to **PDF** and name the file by following format:  
**LAB X – <Student ID>.pdf***

## Section 1.5

1. Hiện thực hóa mô hình trong ví dụ 5.3.1.2, tuy nhiên thay bằng điều kiện sau:  $\text{sells} \leq \text{products} \leq \text{sells} + [77 + 10]$

- Điều kiện:  $\text{sells} \leq \text{products} \leq \text{sells} + 87$

- Source code:

```
/*#####  
# University of Information Technology #  
# IT007 Operating System             #  
# Pham Mai Dung, 19520477            #  
# File: Bai01.c                      #  
#####*/  
  
#include<stdio.h>  
#include<semaphore.h>  
#include<pthread.h>  
  
int sell = 0, product = 0;  
sem_t sem1, sem2;  
  
void *processA(void*mess)  
{  
    while(1)  
    {  
        sem_wait(&sem1);  
        sell++;  
        printf("SELLS = %d\n", sell);  
        sem_post(&sem2);  
    }  
}  
  
void *processB(void*mess)  
{  
    while(1)  
    {  
        sem_post(&sem1);  
        product++;  
        printf("PRODUCTS = %d\n", product);  
        sem_wait(&sem2);  
    }  
}  
  
int main()  
{  
    pthread_t A, B;  
    sem_init (&sem1, 0, 0);  
    sem_init (&sem2, 0, 87);  
    pthread_create (&A, NULL, &processA, NULL);  
    pthread_create (&B, NULL, &processB, NULL);  
    while(1){}  
    return 0;  
}
```

Hình 1.a \_ Source code chạy 2 chương trình “bán” và “mua” song song có đồng bộ

- Kết quả:

```
520477@dung19520477-VirtualBox: ~/lab5-os
PRODUCTS = 11350
PRODUCTS = 11351
PRODUCTS = 11352
PRODUCTS = 11353
PRODUCTS = 11354
SELLS = 11267
SELLS = 11268
SELLS = 11269
SELLS = 11270
SELLS = 11271
SELLS = 11272
SELLS = 11273
SELLS = 11274
SELLS = 11275
SELLS = 11276
SELLS = 11277
SELLS = 11278
SELLS = 11279
SELLS = 11280
SELLS = 11281
SELLS = 11282
SELLS = 11283
SELLS = 11284
SELLS = 11285
SELLS = 11286
SELLS = 11287
SELLS = 11288
SELLS = 11289
SELLS = 11290
SELLS = 11291
SELLS = 11292
```

SELLS = 11334	PRODUCTS = 11429
SELLS = 11335	PRODUCTS = 11430
SELLS = 11336	PRODUCTS = 11431
SELLS = 11337	PRODUCTS = 11432
SELLS = 11338	PRODUCTS = 11433
SELLS = 11339	PRODUCTS = 11434
SELLS = 11340	PRODUCTS = 11435
SELLS = 11341	PRODUCTS = 11436
SELLS = 11342	PRODUCTS = 11437
SELLS = 11343	PRODUCTS = 11438
SELLS = 11344	PRODUCTS = 11439
SELLS = 11345	PRODUCTS = 11440
SELLS = 11346	PRODUCTS = 11441
SELLS = 11347	PRODUCTS = 11442
SELLS = 11348	PRODUCTS = 11443
SELLS = 11349	SELLS = 11356
SELLS = 11350	SELLS = 11357
SELLS = 11351	SELLS = 11358
SELLS = 11352	SELLS = 11359
SELLS = 11353	SELLS = 11360
SELLS = 11354	SELLS = 11361
SELLS = 11355	SELLS = 11362
PRODUCTS = 11355	SELLS = 11363
PRODUCTS = 11356	SELLS = 11364
PRODUCTS = 11357	SELLS = 11365
PRODUCTS = 11358	SELLS = 11366
PRODUCTS = 11359	SELLS = 11367
PRODUCTS = 11360	SELLS = 11368
PRODUCTS = 11361	SELLS = 11369
PRODUCTS = 11362	SELLS = 11370
PRODUCTS = 11363	SELLS = 11371

Hình 1.b, c, d \_ Kết quả thực thi chương trình trên

- Giải thích:
  - + Điều kiện nêu trên đã thỏa mãn.
  - + Khi `product++` thêm 87 được 11354 thì nhường cho process A chạy, điều kiện lúc này là nếu `sells <= products` thì `sell++`, `sell++` lên 87 thì gặp đk `products <= sells` (bán hết sản phẩm), process A nhường cho B tiếp tục sản xuất sản phẩm.

2. Cho một mảng a được khai báo như một mảng số nguyên có thể chứa n phần tử, a được khai báo như một biến toàn cục. Viết chương trình bao gồm 2 thread chạy song song (file đề bài).

\* Chương trình chưa đồng bộ:

- Source code:

```
/*#####  
# University of Information Technology #  
# IT007 Operating System             #  
# Pham Mai Dung, 19520477            #  
# File: Bai02_1.c                    #  
#####*/  
  
#include<stdio.h>  
#include<semaphore.h>  
#include<pthread.h>  
#include<time.h>  
#include<stdlib.h>  
  
int a[10];  
const int n = 10;  
int count = 0;  
  
void AddElements(int a[], int n)  
{  
    srand((int)time(0));  
    int i;  
    for(i= 0; i < n; i++)  
    {  
        a[i] = rand() % 88;//random in [0, 87]  
    }  
    count++;  
}
```

```

        printf("Amount of elements added = %d\n", count);
    }

void OutputElements(int a[], int n)
{
    count--;
    int i;
    for(i = 0; i < n - 1; i++)
    {
        a[i] = a[i + 1];
    }
    if(count == 0)
    {
        printf("Nothing in array a\n");
    }
    else if(count > 0)
    {
        printf("Amount of elements taken = %d\n", count);
    }
}

void *processA(void*mess)
{
    while(1)
    {
        OutputElements(a, n);
    }
}

void *processA(void*mess)
{
    while(1)
    {
        OutputElements(a, n);
    }
}

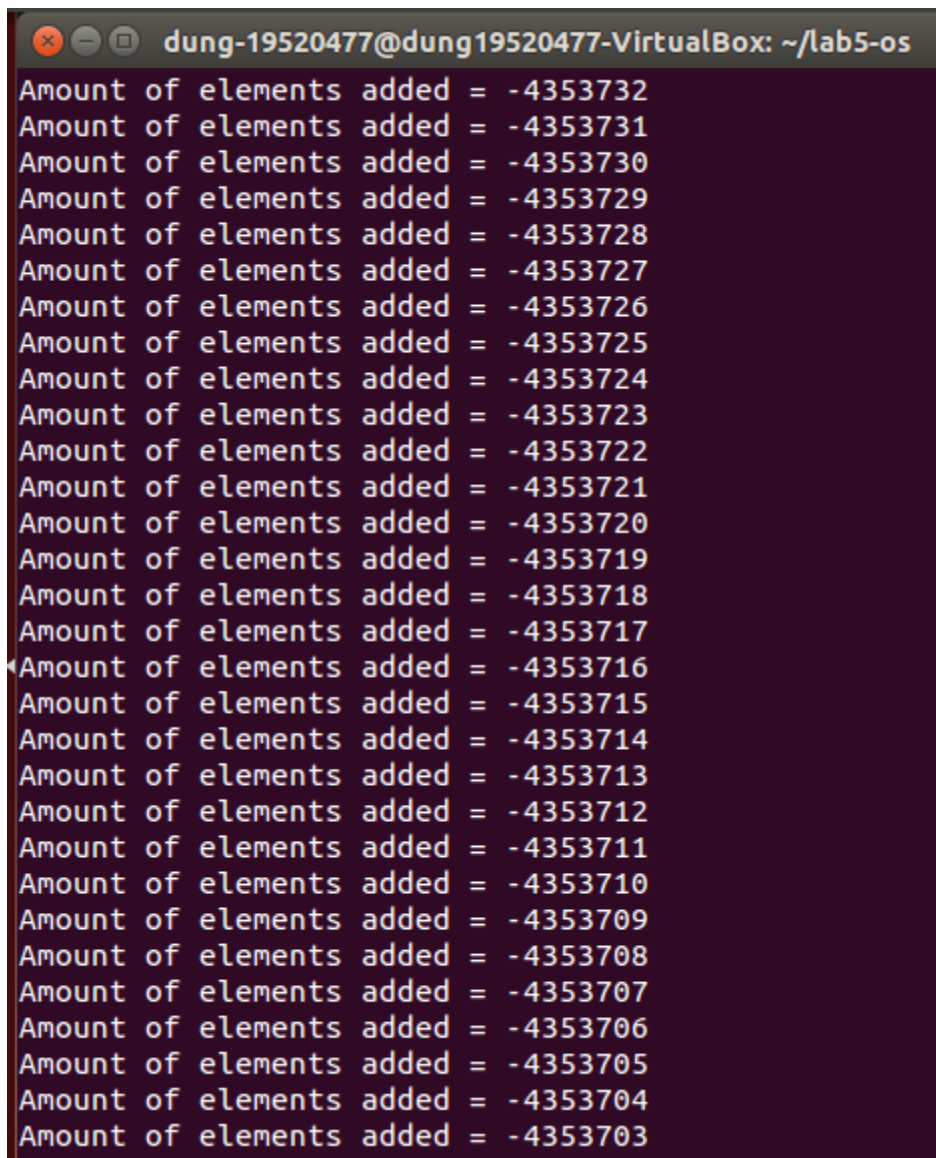
void *processB(void*mess)
{
    while(1)
    {
        AddElements(a, n);
    }
}

int main()
{
    pthread_t A, B;
    pthread_create (&A, NULL, &processA, NULL);
    pthread_create (&B, NULL, &processB, NULL);
    while(1){}
    return 0;
}

```

Hình 2.1.a \_ Source code chương trình chạy song song thêm vào và lấy ra phần tử của mảng (chứa đồng bộ)

- Kết quả:



```
dung-19520477@dung19520477-VirtualBox: ~/lab5-os
Amount of elements added = -4353732
Amount of elements added = -4353731
Amount of elements added = -4353730
Amount of elements added = -4353729
Amount of elements added = -4353728
Amount of elements added = -4353727
Amount of elements added = -4353726
Amount of elements added = -4353725
Amount of elements added = -4353724
Amount of elements added = -4353723
Amount of elements added = -4353722
Amount of elements added = -4353721
Amount of elements added = -4353720
Amount of elements added = -4353719
Amount of elements added = -4353718
Amount of elements added = -4353717
Amount of elements added = -4353716
Amount of elements added = -4353715
Amount of elements added = -4353714
Amount of elements added = -4353713
Amount of elements added = -4353712
Amount of elements added = -4353711
Amount of elements added = -4353710
Amount of elements added = -4353709
Amount of elements added = -4353708
Amount of elements added = -4353707
Amount of elements added = -4353706
Amount of elements added = -4353705
Amount of elements added = -4353704
Amount of elements added = -4353703
```

Hình 2.1.b \_ Kết quả khi chạy chương trình chưa đồng bộ

\* Chương trình khi đồng bộ:

- Source code:



```

/*#####
# University of Information Technology #
# IT007 Operating System             #
# Pham Mai Dung, 19520477            #
# File: Bai02.c                      #
#####*/

#include<stdio.h>
#include<semaphore.h>
#include<pthread.h>
#include<time.h>
#include<stdlib.h>

int a[10];
const int n = 10;
int count = 0;
sem_t sem1, sem2;

void AddElements(int a[], int n)
{
    srand((int)time(0));
    int i;
    for(i= 0; i < n; i++)
    {
        a[i] = rand() % 88;//random in [0, 87]
    }
    count++;
    printf("Amount of elements added = %d\n", count);
}

void OutputElements(int a[], int n)
{
    count--;
    int i;
    for(i = n - 1; i > 0; i--)
    {
        a[i] = a[i - 1];
    }
    if(count == 0)
    {
        printf("Nothing in array a\n");
    }
    else if(count > 0)
    {
        printf("Amount of elements taken = %d\n", count);
    }
}

void *processA(void*mess)
{
    while(1)
    {
        sem_wait(&sem1);
        OutputElements(a, n);
    }
}

```

```

        sem_post(&sem2);
    }
}

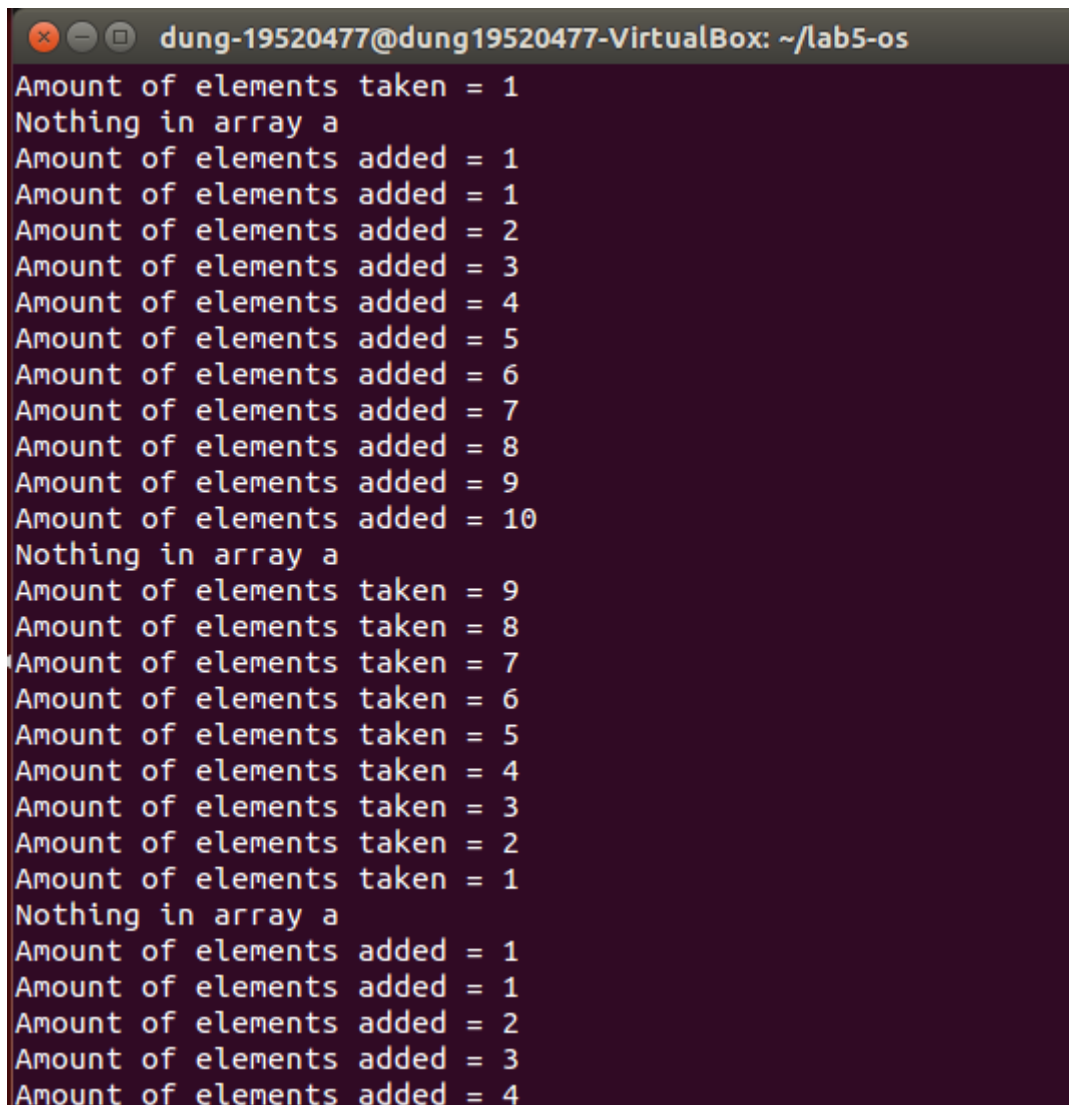
void *processB(void *mess)
{
    while(1)
    {
        sem_post(&sem1);
        AddElements(a, n);
        sem_wait(&sem2);
    }
}

int main()
{
    pthread_t A, B;
    sem_init (&sem1, 0, 0);
    sem_init (&sem2, 0, n);
    pthread_create (&A, NULL, &processA, NULL);
    pthread_create (&B, NULL, &processB, NULL);
    while(1){}
    return 0;
}

```

Hình 2.2.a \_ Source code đã đồng bộ

- Kết quả:



```
dung-19520477@dung19520477-VirtualBox: ~/lab5-os
Amount of elements taken = 1
Nothing in array a
Amount of elements added = 1
Amount of elements added = 1
Amount of elements added = 2
Amount of elements added = 3
Amount of elements added = 4
Amount of elements added = 5
Amount of elements added = 6
Amount of elements added = 7
Amount of elements added = 8
Amount of elements added = 9
Amount of elements added = 10
Nothing in array a
Amount of elements taken = 9
Amount of elements taken = 8
Amount of elements taken = 7
Amount of elements taken = 6
Amount of elements taken = 5
Amount of elements taken = 4
Amount of elements taken = 3
Amount of elements taken = 2
Amount of elements taken = 1
Nothing in array a
Amount of elements added = 1
Amount of elements added = 1
Amount of elements added = 2
Amount of elements added = 3
Amount of elements added = 4
```

Hình 2.2.b \_ Kết quả chương trình đã đồng bộ

- Giải thích: khi dùng semaphore đồng bộ, process B thực thi trước thêm vào cho các phần tử mảng a các giá trị, biến count được đếm thêm 1 lần sau mỗi lần mảng a được thêm, sau khi thêm phần tử thì process A mới được thực thi và lấy từng phần tử ra, biến count giảm dần. Khi không còn phần tử nào trong mảng a thì thông báo “nothing in array a”. Khi chưa đồng bộ, chỉ chạy 1 trong 2 process, biến count cũng tăng hoặc giảm liên tục mà không được đồng bộ hóa.

### 3. Hiện thực mô hình trên C trong hệ điều hành Linux và nhận xét kết quả.

- Source code:

```
/*#####  
# University of Information Technology #  
# IT007 Operating System             #  
# Pham Mai Dung, 19520477            #  
# File: Bai03.c                      #  
#####*/  
  
#include<stdio.h>  
#include<pthread.h>  
#include<semaphore.h>  
  
int x = 0;  
  
void *processA(void*mess)  
{  
    while(1)  
    {  
        x += 1;  
        if(x == 20)  
        { x = 0; }  
        printf("xA = %d\n",x);  
    }  
}  
  
void *processB(void*mess)  
{  
    while(1)  
    {  
        x += 1;  
        if(x == 20)  
        { x = 0; }  
        printf("xB = %d\n",x);  
    }  
}  
  
int main()  
{  
    pthread_t A, B;  
    pthread_create (&A, NULL, &processA, NULL);  
    pthread_create (&B, NULL, &processB, NULL);  
    while(1){}  
    return 0;  
}
```

Hình 3.a \_ Source code theo mô hình

- Kết quả:

```
520477@dung19520477-VirtualBox: ~/lab5-os
xB = 17
xB = 18
xB = 19
xB = 0
xB = 1
xB = 2
xB = 3
xB = 4
xB = 5
xB = 6
xB = 7
xB = 8
xB = 9
xB = 10
xB = 11
xB = 12
xB = 13
xB = 14
xB = 15
xB = 16
xB = 17
xB = 18
xA = 14
xA = 19
xA = 0
xA = 1
xA = 2
xA = 3
xA = 4
xA = 5
xA = 6
```

Hình 3.b \_ Kết quả thực thi chương trình trên

- Nhận xét: khi process A thực thi(x++), giá trị của x được nạp vào thanh ghi, khi hết timeline chuyển qua cho process B sử dụng CPU nạp giá trị của biến x tăng lên và in ra màn hình, sau đó process B hết timeline nên chuyển qua cho process A, process A tăng lên giá trị cũ của x (có thể) nhưng sau đó khi process B nhường lại CPU thì thanh ghi đã ghi lại giá trị tăng x của B, process A chưa kịp nhận nên vẫn tăng giá trị của x cũ lên và in ra ngoài màn hình, sau đó mới nhận x mới và tiếp tục tăng x.

#### 4. Đồng bộ với mutex để sửa lỗi bất hợp lý trong kết quả của mô hình Bài 3.

- Source code:

```
/*#####  
# University of Information Technology #  
# IT007 Operating System             #  
# Pham Mai Dung, 19520477            #  
# File: Bai04.c                      #  
#####*/  
  
#include<stdio.h>  
#include<pthread.h>  
#include<semaphore.h>  
  
int x = 0;  
pthread_mutex_t mutex;  
  
void *processA(void*mess)  
{  
    while(1)  
    {  
        pthread_mutex_lock(&mutex);  
        x += 1;  
        if(x == 20)  
        { x = 0; }  
        printf("xA = %d\n",x);  
        pthread_mutex_unlock(&mutex);  
    }  
}  
  
void *processB(void*mess)  
{  
    while(1)  
    {  
        pthread_mutex_lock(&mutex);  
        x += 1;  
        if(x == 20)  
        { x = 0; }  
        printf("xB = %d\n",x);  
        pthread_mutex_unlock(&mutex);  
    }  
}  
  
int main()  
{  
    pthread_mutex_init(&mutex, NULL);  
    pthread_t A, B;  
    pthread_create (&A, NULL, &processA, NULL);  
    pthread_create (&B, NULL, &processB, NULL);  
    while(1){}  
    return 0;  
}
```

Hình 4.a \_ Source code đồng bộ với mutex

- Kết quả:

```
520477@dung19520477-VirtualBox: ~/lab5-os
xB = 0
xB = 1
xB = 2
xB = 3
xB = 4
xB = 5
xB = 6
xB = 7
xB = 8
xB = 9
xB = 10
xA = 11
xA = 12
xA = 13
xA = 14
xA = 15
xA = 16
xA = 17
xA = 18
xA = 19
xA = 0
xA = 1
xA = 2
xA = 3
xA = 4
xA = 5
xA = 6
xA = 7
xA = 8
xA = 9
xA = 10
```

Hình 4.b \_ Kết quả

- Giải thích: Process A đã cập nhật được giá trị x mới mà process B trước đó đã lưu vào thanh ghi và tiếp tục tăng giá trị x (đã đồng bộ).