

OS 期末群組計畫:Swimming pool Problem

組員:陳逸夫、江明達

1. 內容簡介:

問題內容為泳客進出泳池都必須使用 room 更衣且要使用 basket 放衣服，實作出同步機制有關游泳池的運行並且利用 Poisson process，游泳池運作包括進入和離開，解決 basket 及 room 的同步問題並避免 deadlock，且動態生成 customers，且可動態配置 basket 和 room 的數量

2. 組員分工:

陳逸夫: multithread 程式設計及防止 deadlock 機制及 Poisson process。

江明達: 動態擴充的設計和書面報告。

3. Thread 類型及功能

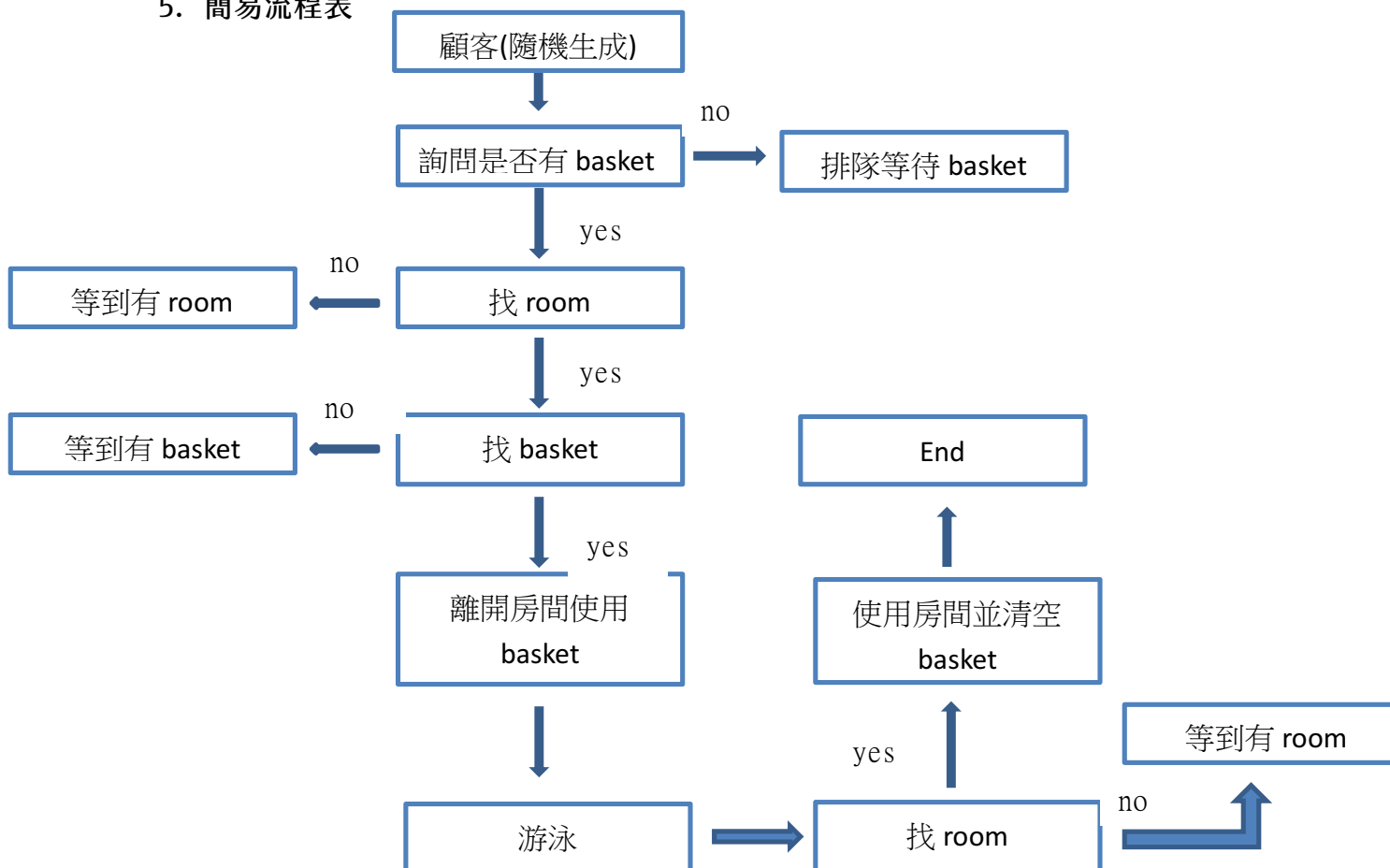
1. 顧客:動態產生人員並且依次去使用 basket 及 Room

4. Multi-Thread 的共有資源與共享變數

1. Basket:動態產生籃子個數用來放置衣服，進入更衣間前必須取得。

2. Room:動態產生更衣間個數，提供給進出泳池顧客使用。

5. 簡易流程表



6. 程式碼

1.Demo.java

```
package os;
import static java.lang.Math.random;
import java.util.*;
import java.util.Scanner;
public class Demo {
    public static int[] ROOMS = new int[5];
    public static int[] BASKETS = new int[10];
    public static int basket=0;
    public static void main(String[] args) throws
InterruptedException {
        int num1;
        int num2;
        int customers;

        java.util.Scanner scanner = new java.util.Scanner(System.in);
        System.out.println("ENTER ROOMS NUMBER:");
        num1 = scanner.nextInt();
        System.out.println("ENTER BASKETS NUMBER:");
        num2 = scanner.nextInt();
        System.out.println("ENTER CUSTOMERS NUMBER:");
        customers = scanner.nextInt()
        Demo.ROOMS = new int[num1];
        Demo.BASKETS = new int[num2];

        int number = 0;
        LinkedList<InThread> inThreadList = null;
        printlnf(ROOMS,BASKETS);
        while(true) {

            Thread.sleep(getPoissonRandom(4000));
            number++;
            if(number>customers)
                break;
            InThread inThread = new InThread(number);
            inThread.start();

        }
    }
}
```

Shared Variable

```

public static void printInf(int[] ROOMS,int[] BASKETS) {

    System.out.println("ROOMS:");
    for(int num:ROOMS) {
        System.out.print(num + " ");
    }
    System.out.println();

    System.out.println("BASKETS:");
    for(int num:BASKETS) {
        System.out.print(num + " ");
    }
    System.out.println();
}

public static int getPoissonRandom(double mean) {
    Random r = new Random();
    double L = Math.exp(-mean);

    int k = 0;
    double p = 1.0;

    do {
        p = p * r.nextDouble();
        k++;
    } while (p > L);
    return k - 1;
}
}

```

Poisson process 之作法

```

3. In Thread.java
package os;
import java.util.Random;
import static os.Demo.printInf;
public class InThread extends Thread {
    private boolean active;
    private int number;
    private int step;
    private int temp1;

```

```

        private int temp2;

    public InThread(int num) {
        active = true;
        number = num;
        step = 0;
    }

    public void setActive(boolean active) {
        this.active = active;
    }

    public boolean isActive() {
        return active;
    }

    static private Object obj = new Object();
    public static void staticWait() {
        synchronized (obj) {
            try {
                obj.wait();
            } catch (Exception e) {}
        }
    }

    public static void staticNotify() {
        synchronized (obj) {
            obj.notify();
        }
    }

    public static void leave(){
        Demo.basket--;
        staticNotify();
    }

    public void run () {
        while(Demo.basket > Demo.BASKETS.length-1){
System.out.println("No BASKET! NO."+ number +" SEND TO WAITLIST ");
            staticWait();
        }
    }
}

```

防止 Deadlock 機制

```
    }
    Demo.basket++;
while(step<5) {
    switch(step) {
case 0:{
    System.out.println("NO."+ number + " Finding ROOM");
    for(int
avalivable=0;avalivable<Demo.ROOMS.length;avalivable++) {
        if(Demo.ROOMS[avalivable] == 0) {
            Demo.ROOMS[avalivable] = number;
            temp1 = avalivable+1;
            step++;

            System.out.println("USE NO."+ temp1 + " ROOM");
            break;
        }
        if(avalivable==Demo.ROOMS.length-1)
            System.out.println("!!!NO Room for "+number+"!!!");
    }
    break;
}

case 1:{
    System.out.println("No."+number + " Finding BASKET");
    for(int avalivable=0; avalivable<Demo.BASKETS.length;
avalivable++) {
        if(Demo.BASKETS[avalivable] == 0) {
            Demo.BASKETS[avalivable] = number;
            temp2 = avalivable+1;
            Demo.ROOMS[temp1-1] = 0;
            System.out.println("NO."+ temp1 + " ROOM
EMPTY");

            step++;
            System.out.println("USE NO."+ temp2 + " BASKET");
            break;
        }
        if(avalivable==Demo.BASKETS.length-1)
            System.out.println("!!!NO Basket for "+number+"!!!");
    }
}
```

Critical Section

```

        break;
    }
case 2:{
    System.out.println("NO."+number + " SWIMMING");
    Step++;
    break;
}
case 3:{
    System.out.println("NO."+number + " Finding ROOM(2ND)");
    for(int i=0; i<Demo.ROOMS.length; i++) {
        if(Demo.ROOMS[i] == 0) {
            Demo.ROOMS[i] = number;
            temp1 = i+1;
            step++;
        }
        System.out.println("USE NO."+ temp1 + " ROOM");
        Demo.BASKETS[temp2-1] = 0;
        leave();

        System.out.println("NO."+ temp2 + " BASKET EMPTY");
        break;
    }
    if(i==Demo.ROOMS.length-1)
        System.out.println("!!!NO Room for "+number + "!!!");
    break;
}
case 4:{
    System.out.println("NO."+ number + " LEAVE POOL");
    System.out.println("NO."+ temp1 + " ROOM EMPTY");
    tep++;
    Demo.ROOMS[temp1-1] = 0
    break;
}
}

try {
    Thread.currentThread();
    Thread.sleep(Demo.getPoissonRandom(number+10)*300);
}

```

```
catch(InterruptedException e) {  
    e.printStackTrace();  
}  
}  
}
```

7. 參考資料及程式語言及編譯器

1.github

2.java

3.netbeans

4.Poisson process:

<http://stackoverflow.com/questions/9832919/generate-poisson-arrival-in-java>

5.Java multithread

http://www.tutorialspoint.com/java/java_multithreading.htm