به نام خدا

ثنا موسوی-99243070

کلاس FIFO

در این کلاس ابتدا یک ارایه از جنس integer تعریف شده است که شماره‌ی مشتری ها در ان قرار میگیرند.همچنین شامل متغیری گلوبال page fault است تا مقادیر را همواره داشته باشیم در ادامه تابع add_page که شماره‌ی مشتری جدید را میگیرد و بررسی میکند اگر این شماره در ارایه نباشد، بررسی میکند ایا با توجه به capacity که در ابتدا داده ایم میتوانیم مشتری جدیدی اضافه کنیم یا نه؟

اگر تعداد مشتری ها کمتر بود صرفا این مشتری را اضافه میکنیم و تعداد page faultها اضافه میشود و تعدا سایز هم زیاد میشود و در غیر اینصورت متغیر front یکی به جلو میرود یعنی عملا اولین کسی که امده حذف میشود و از اونور صف هم ما سایز رو زیاد میکنیم پس داده ای رو از دست نمیدهیم

```
String add_page(int customer_id) {
    if (!findCustomer(customer_id)) {
        if (size==capacity) {
            front=(front+1)%capacity;
        }
        items[size]=customer_id;
        size=(size+1)%capacity;
        page_faults++;
    }
    return Arrays.toString(items).replaceAll("[\\[\\]]","");
}
```

کلاس LRU

در این کلاس یک ارایه از جنس int و hash map<integer, integer> وجود دارد که برای نگهداری مشتری ها استفاده میشود

در این متود در صورتی که مشتری ای با این عدد وجود نداشته باشد ،ابتدا بررسی میکنیم که ظرفیت داریم یا نه؟اگر ظرفیت داشته باشیم این مشتری را به ته ارایه و map اضافه میکنیم.و یک متغیر i داریم که هر چه مقدار ان بیشتر باشد یعنی اخیرا استفاده شده است و در مپ ان را هم قرار میدهیم. و در صورتی که ارایه ما پر باشد اندیس کسی که دیر تر از همه استفاده رده است را با کمک mapبه دست میاوریم و این عنصر را در ارایه جایگزین میکنیم.

```
String add_page(int customer_id) {
    if (!findCustomer(customer_id)) {
        page_faults++;
        if (size== capacity) {
            Map.Entry<Integer, Integer> min =
Collections.min(indexes.entrySet(), new Comparator<Map.Entry<Integer,
Integer>>() {
                public int compare(Map.Entry<Integer, Integer> entry1,
Map.Entry<Integer, Integer> entry2) {
                    return entry1.getValue().compareTo(entry2.getValue());
                }
            });
            pages_in[findCustomerIndex(min.getKey())]=customer_id;
            indexes.remove(min.getKey());
        }
```

```
        else {
            pages_in[size] = customer_id;
            size = size + 1;
        }
    }
    indexes.put(customer_id,i);
    i++;
    return Arrays.toString(pages_in).replaceAll("[\\[\\]]", "");
}
```

کلاس second chance

ابتدا یک کلاس node تعریف کرده ایم شامل دو متغیر reference bit,value و در این کلاس ارایه ای از node داریم.

وقتی یک مشتری جدید میاید 3 حالت داریم

اگر مشتری جدید است و ظرفیت تکمیل نشده است با bit=0 درون ارایهقرار میگیرد

اگر مشتری جدیدی بیاید و size=capacity باشد در اینصورت درون ارایه جست و جو میکنیم و اگر کسی bit=0 باشد ان را ریموو میکنیم و نفر جدید را اضافه میکنیم و در غیر اینصورت یعنی اگر فردی با bit=1 وجود داشته باشد یک شانس دوباره به ان میدهیم و به انتهای لیست اضافه میکنیم.

اگر مشتری وجود داشته باشد و bit=0 باشد bit ان را به 1 تغییر میدهیم یعنی به ان اشاره شده است

```
{String add_page(int customer_id) {
    int inQueue = findCustomerIndex(customer_id);
    if (inQueue != -1) {
        items[inQueue].setReference_bit(1);
    }
    /*
    by this we mean the id is not in the restaurant
     */
    else {
        page_faults++;
        if (capacity == size) {
            int shift = 0;
            for (int j = front; ; j++) {
                if (items[j%capacity].getReference_bit() == 0) {
                    items[j%capacity].setData(customer_id);
                    break;
                } else {
                    shift++;
                    items[j%capacity].setReference_bit(0);
                }
            }

            front = (front + shift+1) % capacity;

        } else {
            items[size] = new node(customer_id, 0);
            size = size + 1;
        }
    }
```

```
    /*there is no customer by this id, then we add this and the bit is 0*/
    //indexes.add(new node(customer_id,0));
    return
Arrays.stream(items).filter(Objects::nonNull).map(node::getData).collect(Coll
ectors.toList()).toString().replaceAll("[\\[\\]]", "");

}

}
```

برای بخش thread

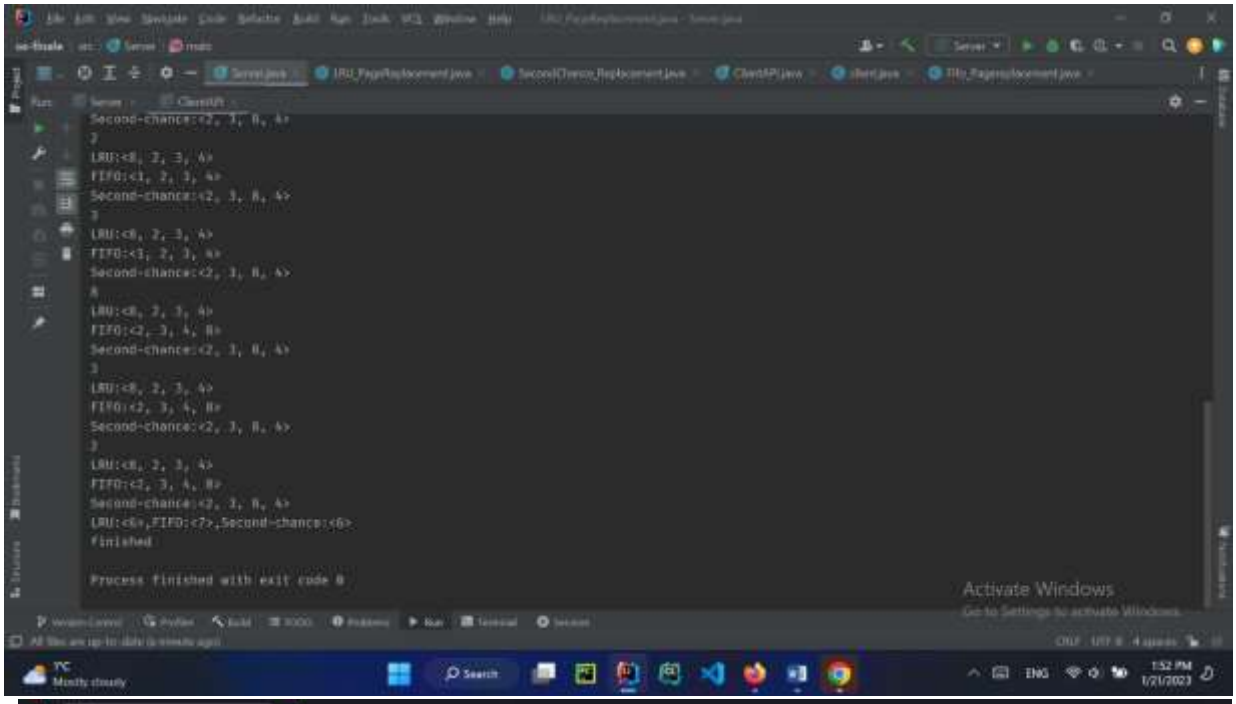
یک نخ از کلاس کلاینت میسازیم و بعد از اولین داده ترد را استارت میکنیم و در ادامه داده ها را در یک بافر ذخیره میکنیم تا از دست نرود و در اخر منتظر میمانیم تا کار ترد تمام شود و برنامه را خاتمه میدهیم

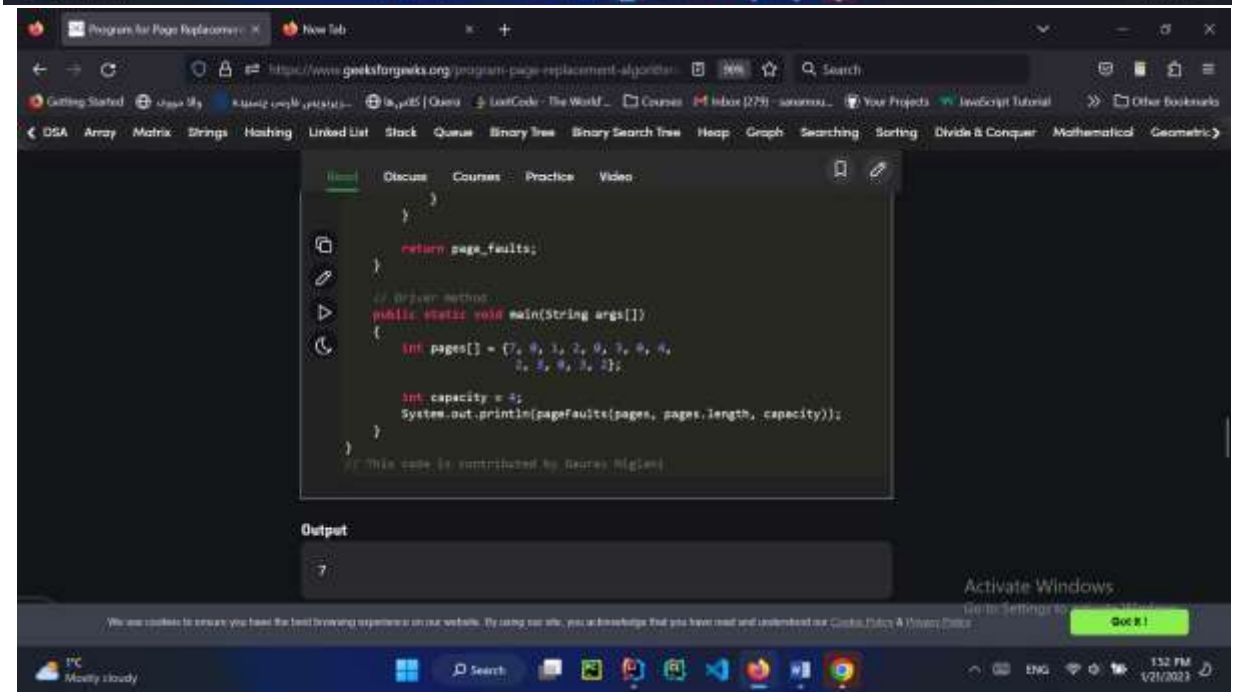
در متد ران هم تا زمانیکه 0 وارد نشده است صف ها را پرینت میکنیم

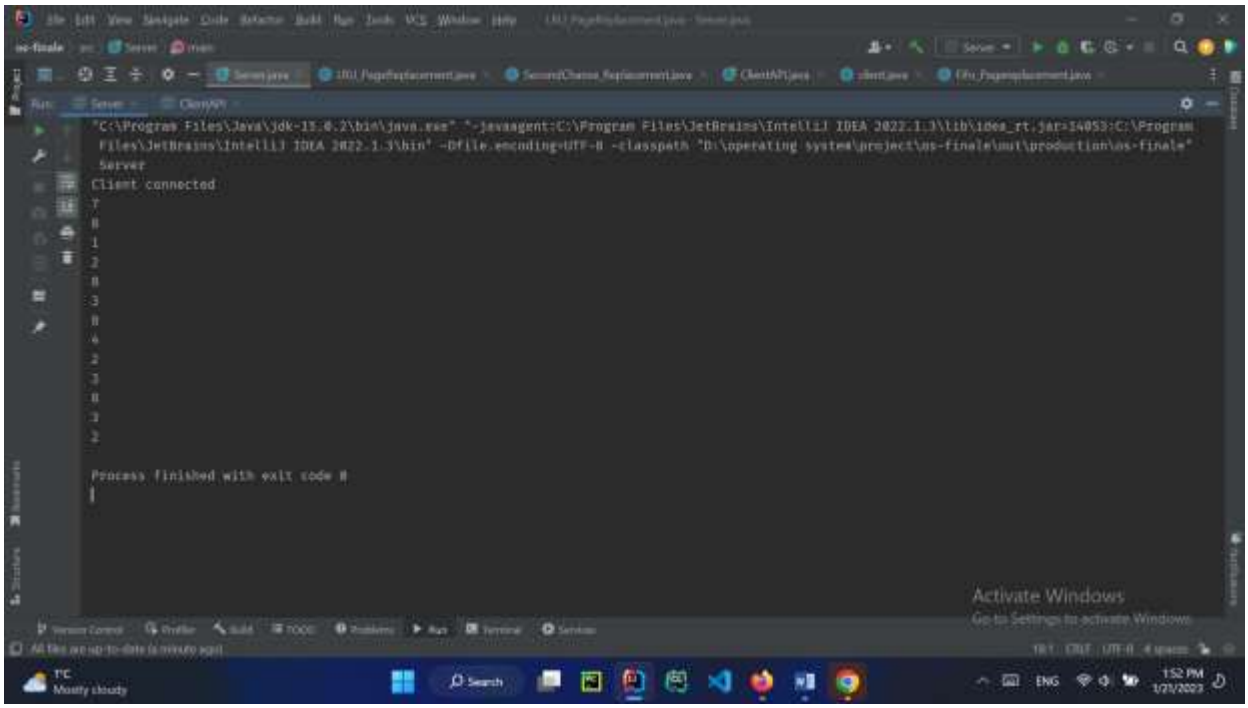
```
Thread th=new Thread(new client(line));
line=din.readInt();
data.add(line);
th.start();
while (line!=0) {
    line = din.readInt();
    data.add(line);
}
try {
    th.join();
} catch (InterruptedException e) {
    throw new RuntimeException(e);
}
close();
```

```
@Override
public void run() {
    loop:
    while (true) {
        int customer_id = ClientAPI.send_data();
        switch (customer_id) {
            case 0:
                System.out.printf("LRU:<%d>,FIFO:<%d>,Second-chance:<%d>%n", lru.getPage_faults(),
                        fifo.getPage_faults(), secondChance.getPage_faults());
                break loop;
            default:
                System.out.println(customer id);
                System.out.printf("LRU:<%s>\nFIFO:<%s>\nSecond-chance:<%s>%n", lru.add_page(customer_id),
                        fifo.add_page(customer_id), secondChance.add_page(customer_id));
                break;

        }
    }
}
```


Second-chance:<2, 3, 8, 4>
3
LRU:<8, 2, 3, 4>
FIFO:<1, 2, 3, 4>
Second-chance:<2, 3, 8, 4>
3
LRU:<8, 2, 3, 4>
FIFO:<1, 2, 3, 4>
Second-chance:<2, 3, 8, 4>
4
LRU:<8, 2, 3, 4>
FIFO:<2, 3, 4, 8>
Second-chance:<2, 3, 8, 4>
3
LRU:<8, 2, 3, 4>
FIFO:<2, 3, 4, 8>
Second-chance:<2, 3, 8, 4>
3
LRU:<8, 2, 3, 4>
FIFO:<2, 3, 4, 8>
Second-chance:<2, 3, 8, 4>
LRU:<6>,FIFO:<7>,Second-chance:<6>
finished
Process finished with exit code 0
Activate Windows


Program for Page Replacement
New Tab
https://www.geeksforgeeks.org/program-page-replacement-algorithm
Search
DSA
Array
Matrix
Strings
Hashing
Linked List
Stack
Queue
Binary Tree
Binary Search Tree
Heap
Graph
Searching
Sorting
Divide & Conquer
Mathematical
Discuss
Courses
Practice
Video
return page_faults;
// Driver method
public static void main(String args[])
int pages[] = {7, 0, 1, 2, 0, 3, 0, 4,
2, 3, 0, 3, 2};
int capacity = 4;
System.out.println(pageFaults(pages, pages.length, capacity));
Output
7
Got it !
Activate Windows


"C:\Program Files\Java\jdk-15.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2022.1.3\lib\idea_rt.jar=14053:C:\Program Files\JetBrains\IntelliJ IDEA 2022.1.3\bin" -Dfile.encoding=UTF-8 -classpath "D:\operating system\project\os-finale\out\production\os-finale" Server
Client connected
7
8
1
2
8
3
8
4
2
3
8
3
2
Process finished with exit code 0
Activate Windows
Go to Settings to activate Windows.
Mostly cloudy
Search
ENG
1:52 PM
1/21/2023