

DẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TINH



Hệ Điều Hành mở rộng (CO201D)

Bài tập lớn

Giả lập môi trường ảo trên CodeRunne và xây dựng testcase CPU Scheduling và Demand Paging

GVHD: Dinh Quang Thịnh
Nguyễn Hoàng Việt - 1814771



Mục lục

1 Moodle	3
1.1 Moodle là gì? [1]	3
1.2 Cài đặt Moodle	3
1.3 Thêm một trang môn học mới	5
1.4 Thêm plugin CodeRunner vào Moodle	6
2 Xây dựng môi trường và build testcase cho CPU Scheduling	9
2.1 FCFS CPU Scheduling	10
2.2 SJF CPU Scheduling (Non-preemptive)	12
2.3 SJF CPU Scheduling (Preemptive)	15
2.4 LRTF Scheduling	18
2.5 Round Robin Scheduling	21
3 Xây dựng môi trường và build testcase cho DemandPaging - Page Replacement	25
3.1 FIFO	25
3.2 Optimal	27
3.3 LRU	30
3.4 Second chance - Clock	32
4 Đánh giá tổng hợp kết quả đạt được	36
Tài liệu tham khảo	37



1 Moodle

1.1 Moodle là gì? [1]

Moodle là một hệ thống quản lý học tập (Learning Management System - LMS hoặc người ta còn gọi là Course Management System hoặc VLE - Virtual Learning Environment) mã nguồn mở (do đó miễn phí và có thể chỉnh sửa được mã nguồn), cho phép tạo các khóa học trên mạng Internet hay các website học tập trực tuyến.

Moodle (viết tắt của Modular Object-Oriented Dynamic Learning Environment) được sáng lập năm 1999 bởi Martin Dougiamas, người tiếp tục điều hành và phát triển chính của dự án. Do không hài lòng với hệ thống LMS/LCMS thương mại WebCT trong trường học Curtin của Úc, Martin đã quyết tâm xây dựng một hệ thống LMS mã nguồn mở hướng tới giáo dục và người dùng hơn. Từ đó đến nay Moodle có sự phát triển vượt bậc và thu hút được sự quan tâm của hầu hết các quốc gia trên thế giới và ngay cả những công ty bán LMS/LCMS thương mại lớn nhất như BlackCT (BlackBoard + WebCT) cũng có các chiến lược riêng để cạnh tranh với Moodle.

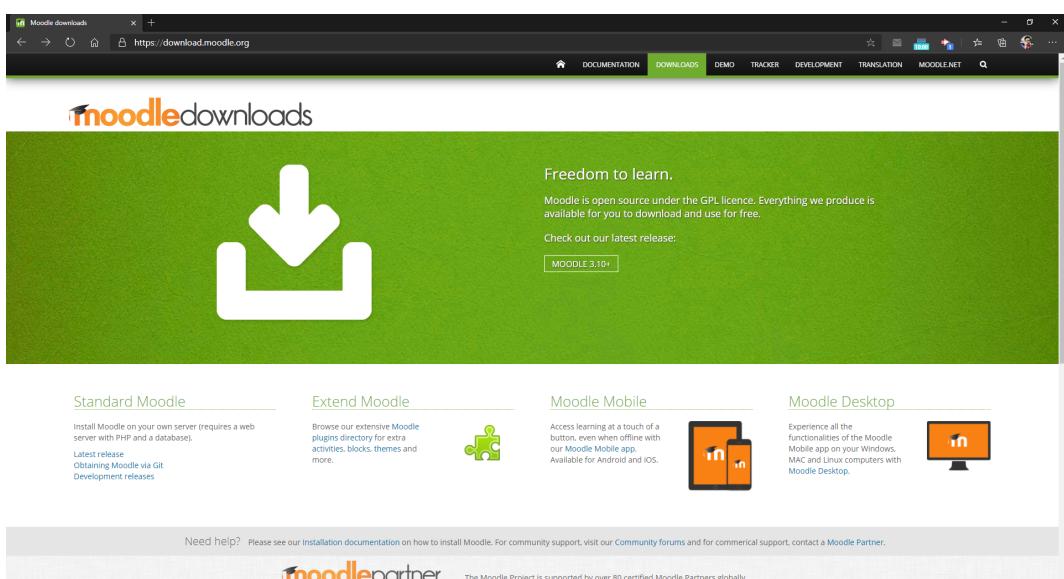
Moodle nổi bật là thiết kế hướng tới giáo dục, dành cho những người làm trong lĩnh vực giáo dục. Moodle rất dễ dùng với giao diện trực quan, giáo viên chỉ mất một thời gian ngắn để làm quen và có thể sử dụng thành thạo. Do thiết kế dựa trên module nên Moodle cho phép bạn chỉnh sửa giao diện bằng cách dùng các theme có sẵn hoặc tạo thêm một theme mới cho riêng mình. Moodle phù hợp với nhiều cấp học và hình thức đào tạo: phổ thông, đại học/cao đẳng, không chính quy, trong các tổ chức/công ty.

Moodle phát triển dựa trên PHP (Ngôn ngữ được dùng bởi các công ty Web lớn như Yahoo, Flickr, Baidu, Digg, CNET) có thể mở rộng từ một lớp học nhỏ đến các trường đại học lớn trên 50 000 sinh viên (ví dụ đại học Open PolyTechnique của Newzealand hoặc sắp tới đây là đại học mở Anh

- Open University of UK, trường đại học cung cấp đào tạo từ xa lớn nhất châu Âu, và đại học mở Canada, Athabasca University). Ta có thể dùng Moodle với các database mã nguồn mở như MySQL hoặc PostgreSQL. Phiên bản 1.7 sẽ hỗ trợ thêm các database thương mại như Oracle, Microsoft SQL để có thêm nhiều cơ hội lựa chọn.

1.2 Cài đặt Moodle

Ta có thể tải Moodle ở [Đây: https://download.moodle.org/](https://download.moodle.org/)



Hình 1: Trang tải về của Moodle



Sau khi tải về ta tiến hành giải nén, chạy "Start Moodle.exe" và điền các thông tin cung như cài đặt những công nghệ yêu cầu:

The screenshot shows the 'Installation' page of Moodle. The URL in the browser is `localhost/user/editadvanced.php?id=2`. The page title is 'Installation'. A note at the top says: 'On this page you should configure your main administrator account which will have complete control over the site. Make sure you give it a secure username and password as well as a valid email address. You can create more admin accounts later on.' Below this, there is a section titled 'General' with the following fields:

- Username: admin
- Choose an authentication method: Manual accounts
- New password: Click to enter text (with a password strength indicator)
- First name: Admin
- Surname: User
- Email address: (empty)
- Email display: Allow everyone to see my email address
- MoodleNet profile: (empty)
- City/town: (empty)
- Select a country: (empty)

Hình 2: Tạo tài khoản admin

The screenshot shows the 'Front page settings' and 'Location settings' pages of Moodle.

New settings - Front page settings

- Full site name: Demo
- Short name for site (eg single word): First site Demo
- Front page summary:
summary
Demo Moodle

A note below the summary area says: 'This summary can be displayed on the front page using the course/site summary block.'

New settings - Location settings

- Default timezone: Australia/Perth
- A note below says: 'This is the default timezone for displaying dates - each user can override this setting in their profile. Cron tasks and other server settings are specified in this timezone. You should change the setting if it shows as "Invalid timezone".'

New settings - Manage authentication

- Self registration: Disable

Hình 3: Front page settings

Sau khi cài đặt thành công ta sẽ có giao diện như sau:



The screenshot shows the Moodle homepage with the following sections:

- Recently accessed courses:** Displays a message "No recent courses".
- Course overview:** Displays a message "No courses".
- Timeline:** Displays a message "No upcoming activities due".
- Private files:** Displays a message "No files available".
- Online users:** Shows 1 online user (Admin User) last 5 minutes.
- Latest badges:** Shows a message "You have no badges to display".
- Calendar:** Shows the month of November 2020.

Hình 4: HomePage

1.3 Thêm một trang môn học mới

Để thêm một trang học mới, ta vào Site Home để tạo trang học mới. sau đó ấn vào "Add a new course"

The screenshot shows the Site Home page with the following sections:

- Available courses:** Contains a button labeled "Add a new course".

Hình 5: Thêm một trang học mới



The screenshot shows the 'Add a new course' form in Moodle. The 'General' section includes fields for Course full name ('Hệ điều hành mở rộng - (CO201D)'), Course short name ('Hệ điều hành'), Course category ('No selection'), Course visibility ('Show'), Course start date ('30 November 2020'), Course end date ('30 November 2021'), and Course ID number. The 'Description' section contains a rich text editor with the text 'Học phần mở rộng của môn Hệ điều hành'.

Hình 6: Diền thông tin của môn học

The screenshot shows the 'Courses' page in Moodle. It lists the course 'Hệ điều hành mở rộng - (CO201D)' with the description 'Học phần mở rộng của môn Hệ điều hành'. There are buttons for 'Add a new course' and 'Courses pending approval'.

Hình 7: Môn học đã được tạo thành công

1.4 Thêm plugin CodeRunner vào Moodle

Dầu tiên ta cần tải các plugins cần thiết ở [link sau](#): **CodeRunner** và **Adaptive adapted for coderunner**

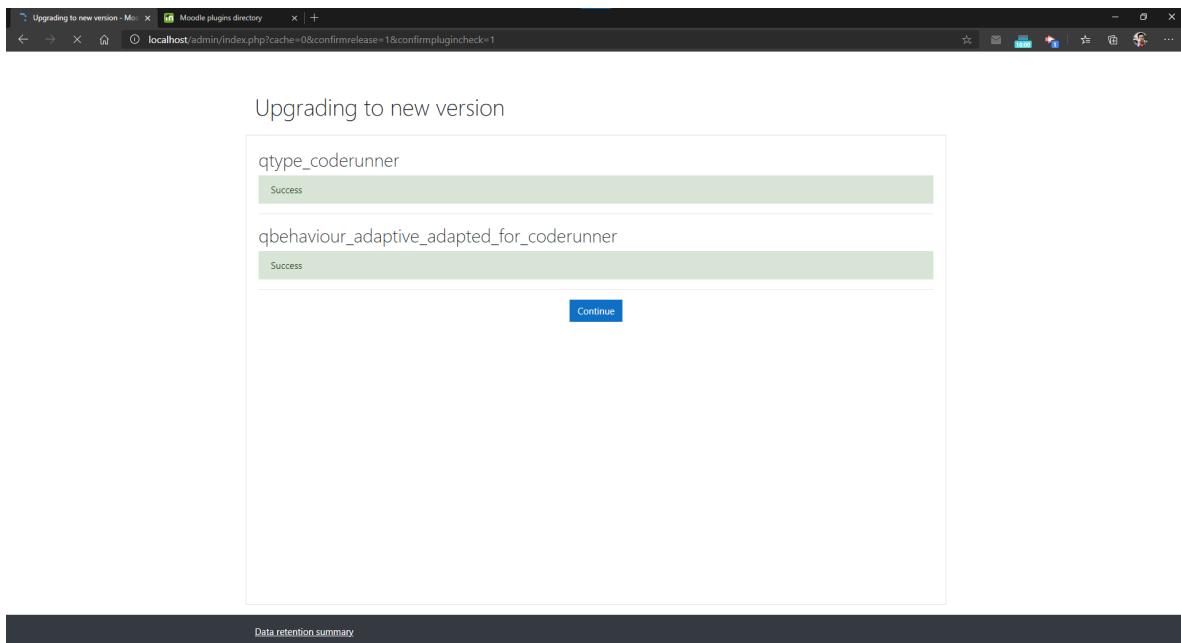


Hình 8: Tải plugin CodeRunner

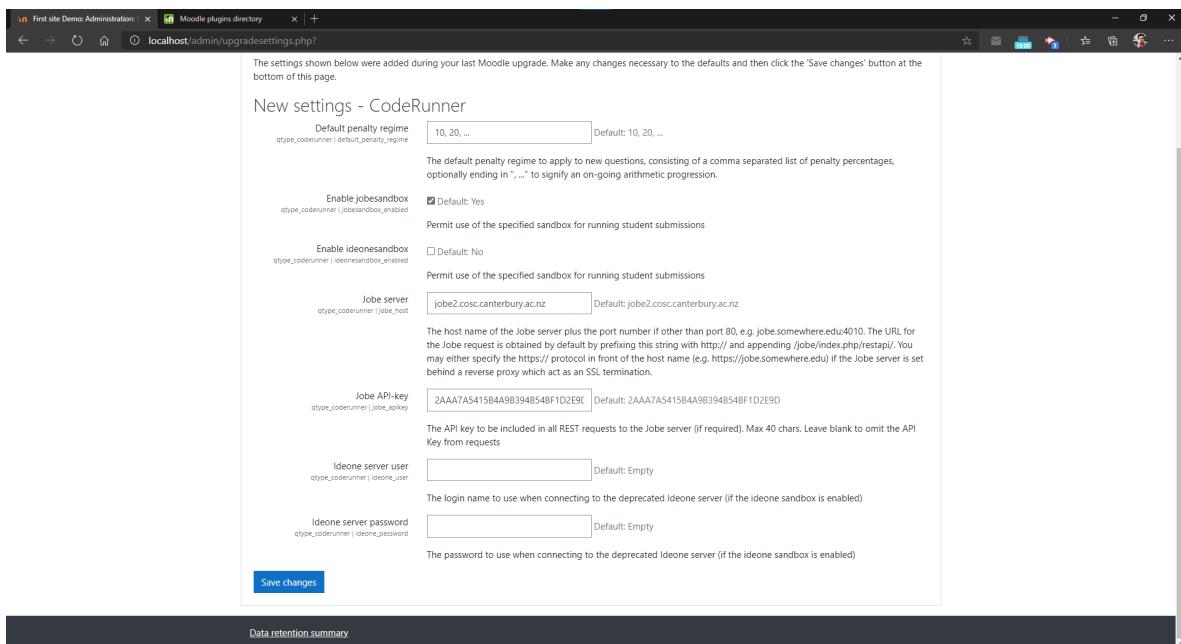
Sau khi tải về ta sẽ được 2 file zip, sau đó ta cần giải nén plugin "CodeRunner" vào folder /question/type/coderunner, còn plugin "Adaptive adapted for coderunner" vào thư mục question/behaviour/adaptive_adapted_for_coderunner. Sau đó ta mở Moodle vào "Site administration" ta sẽ được giao diện như sau:

Hình 9: Cài đặt plugin CodeRunner

Sau đó ta ấn vào "Upgrade Moodle database now" để cài đặt 2 plugin mà ta đã thêm vào.



Hình 10: Cài đặt plugin CodeRunner thành công



Hình 11: Settings cho plugin CodeRunner

Sau khi cài đặt thành công, ta vào site môn học đã tạo từ trước, tạo một quiz. Sau đó ấn vào thêm câu hỏi để kiểm tra xem plugin CodeRunner đã được thêm vào thành công chưa:



The screenshot shows the Moodle quiz editing interface. A modal window titled "Choose a question type to add" is open, displaying various question types: Multiple choice, True/false, Matching, Short answer, Essay, Calculated, Calculated multichoice, Calculated simple, and CodeRunner. The "CodeRunner" option is selected. The main quiz page shows a single question titled "square for python square". The quiz has a maximum grade of 10.00 and a total of 1.00 marks.

Hình 12: Kiểm tra plugin CodeRunner đã cài đặt thành công

2 Xây dựng môi trường và build testcase cho CPU Scheduling

Đối với các giải thuật CPU Scheduling, em sẽ thiết lập những câu quiz yêu cầu tính thời gian đợi trung bình (Average waiting time) và thời gian xoay vòng trung bình (Average turn around time).

The screenshot shows the Moodle quiz editing interface for "CPU Scheduling". The quiz contains five questions, each with a title and a detailed description. The questions are: 1. Programming for FCFS CPU Scheduling (describing the function findavgTime), 2. Programming for SJF CPU Scheduling (Non-preemptive) (describing the function findavgTime), 3. Programming for SJF CPU Scheduling (preemptive) (describing the function findavgTime), 4. Programming for LRTF CPU Scheduling (describing the function findavgTime), and 5. Programming for RoundRobin CPU Scheduling (describing the function findavgTime). Each question is worth 1.00 mark.

Hình 13: Câu quiz dành cho các giải thuật CPU Scheduling



2.1 FCFS CPU Scheduling

Dầu tiên ta tạo một câu quiz mới, sau đó đặt tên là FCFS CPU Scheduling, và viết các mô tả cần thiết:

Editing a CodeRunner question Hộ điều hành: CPU Scheduling +

localhost/question/question.php?id=21&qtype=coderunner&cmid=38&courseid=28&returnurl=%2Fmod%2Fquiz%2Fedit.php%3Fcmid%3D%26amp%3Bcat%3D4%25C2%256amp%3...

First site Demo

Question type details

General

Current category: Default for Hộ điều hành (2) Use this category

Save in category: Default for Hộ điều hành (2)

Question name: Program for FCFS CPU Scheduling

Question text:

```
Viết function findavgTime để xuất ra thời gian đợi trung bình và thời gian xoay vòng trung bình của giải thuật FCFS CPU Scheduling theo template như sau:  
def findavgTime(processes):  
    #T000: write function to calculate avgWaitingTime and avgTurnAroundTime of FCFS algorithm  
    #! DO NOT CHANGE  
    print("Average waiting time = "+ "{:.2f}".format(avgWaitingTime))  
    print("Average turn around time = "+ "{:.2f}".format(avgTurnAroundTime))  
Để đóng với kết quả, in đúng theo kết quả testcase dưới đây, lấy 2 chữ số sau chữ số thập phân (như code mẫu đã sinh ra).
```

Default mark: 1

General feedback:

Hình 14: Mô tả quiz FCFS

Sau đó ta thêm những testcase, hoặc ta có thể thêm code rồi sau khi bấm save, ta dùng kết quả sau khi chạy code để làm expected output

Editing a CodeRunner question Hộ điều hành: CPU Scheduling +

localhost/question/question.php?id=21&qtype=coderunner&cmid=38&courseid=28&returnurl=%2Fmod%2Fquiz%2Fedit.php%3Fcmid%3D%26amp%3Bcat%3D4%25C2%256amp%3...

First site Demo

Test case 1

Standard Input: `findavgTime((0,10),(0,5),(0,8))`

Expected output: `Average waiting time = 8.33
Average turn around time = 16.00`

Extra template data:

Test properties: Use as example, Display Show, Hide rest if fail, Mark 1.000, Ordering 10

Test case 2

Standard Input: `findavgTime((0,7),(0,2),(0,5),(0,8),(0,2),(0,1))`

Expected output: `Average waiting time = 12.67
Average turn around time = 16.83`

Extra template data:

Test properties: Use as example, Display Show, Hide rest if fail, Mark 1.000, Ordering 20

Test case 3

Standard Input: `findavgTime([(0,5),(0,2),(0,6),(0,2),(0,1),(0,7)])`

Expected output: `Average waiting time = 10.00
Average turn around time = 16.00`

Extra template data:

Test properties: Use as example, Display Show, Hide rest if fail, Mark 1.000, Ordering 30

Hình 15: FCFS dụng testcase



```
3     n = len(processes)
4     wt = [0] * n
5     tat = [0] * n
6     completeTime = [0] * n
7
8     # Function to find turn around time
9     for i in range(n):
10         completeTime[i] = max(completeTime[i-1], processes[i][0]) + processes[i][1] if i
11         >0 else processes[i][0] + processes[i][1]
12         tat[i] = completeTime[i] - processes[i][0] # turn around time = completeTime -
13         arrival time
14
15     # finding waiting time
16     for i in range(n):
17         wt[i] = tat[i] - processes[i][1]
18
19     avgWaitingTime = sum(wt) / n
20     avgTurnAroundTime = sum(tat) / n
21
22     #! DO NOT CHANGE
23     print("Average waiting time = " + "{:.2f}".format(avgWaitingTime))
24     print("Average turn around time = " + "{:.2f}".format(avgTurnAroundTime))
```

2.2 SJF CPU Scheduling (Non-preemptive)

Thêm một câu hỏi mới là SJF CPU Scheduling (Non-preemptive)

The screenshot shows the Moodle Question editor interface. On the left, there is a sidebar with navigation links like 'Hệ điều hành', 'Participants', 'Badges', 'Competencies', 'Grades', 'General', 'Topic 1' (which is selected), 'Topic 2', 'Topic 3', 'Topic 4', 'Dashboard', 'Site home', 'Calendar', 'Private files', 'Content bank', and 'Site administration'. The main area has fields for 'Current category' (set to 'Default for Hệ điều hành'), 'Save in category' (dropdown menu), 'Question name' (text input: 'Programming for SJF CPU Scheduling (Non-preemptive)'), 'Question text' (richtext editor with a note about the code provided), 'Default mark' (text input: '1'), 'General feedback' (richtext editor), 'ID number' (text input), and an 'Answer' section (dropdown menu). The URL in the browser bar is 'localhost/question/question.php?returnurl=%2Fmod%2Fquiz%2Fedit.php%3Fcmid%3D3%26amp%3Bcat%3D4%25&cmid=3&id=22'.

Hình 18: SJF Non-preemptive mô tả quiz

Sau đó ta thêm những testcase cần thiết cho quiz này



The screenshot shows a web-based test case editor for a SJF Non-preemptive quiz. On the left, a sidebar lists topics and site administration. The main area is titled 'test cases' and contains three entries:

- Test case 1:** Standard Input: `findavgTime([(2,3),(0,4),(4,2),(5,4)])`; Expected output: `Average waiting time = 2.00`, `Average turn around time = 5.25`.
- Test case 2:** Standard Input: `findavgTime([(3,3),(0,4),(4,2),(5,4),(1,2),(4,5),(2,1),(2,5),(4,5)])`; Expected output: `Average waiting time = 8.33`, `Average turn around time = 11.78`.
- Test case 3:** Standard Input: `findavgTime([(3,3),(0,4),(4,2),(5,4),(1,2),(4,5),(2,1),(2,5),(4,5)])`.

Each test case has a 'Test properties' section with checkboxes for 'Use as example', 'Display', 'Show', 'Hide rest if fail', 'Mark', and 'Ordering'.

Hình 19: SJF Non-preemptive testcase

Chạy thử lại quiz xem đã chính xác chưa

The screenshot shows a quiz page for the SJF Non-preemptive algorithm. The question details are:

Question 1
Not complete
Marked out of 1.00

Viết function findavgTime để xuất ra thời gian đợi trung bình và thời gian xoay vòng trung bình của giải thuật SJF CPU Scheduling (Non-preemptive) theo template như sau:

```
def findavgTime(processes):
    #TODO: write functions to calculate avgWaitingTime and avgTurnAroundTime of SJF Algorithme (Non-preemptive)

    #! DO NOT CHANGE
    print("Average waiting time = "+ "{:.2f}".format(avgWaitingTime))
    print("Average turn around time = "+ "{:.2f}".format(avgTurnAroundTime))
```

Hàm findavgTime được truyền vào là một list của các tuple 2 phần tử, phần tử thứ nhất sẽ là arrival time của process, phần tử thứ hai sẽ là burst time của process.

Để đúng với kết quả, in đúng theo kết quả testcase dưới đây, lấy 2 chữ số sau chữ số thập phân (như code mẫu đã sinh ra).

For example:

Test	Result
<code>findavgTime([(2,3),(0,4),(4,2),(5,4)])</code>	<code>Average waiting time = 2.00</code> <code>Average turn around time = 5.25</code>
<code>findavgTime([(3,3),(0,4),(4,2),(5,4),(1,2),(4,5),(2,1),(2,5),(4,5)])</code>	<code>Average waiting time = 8.33</code> <code>Average turn around time = 11.78</code>

Answer: (penalty regime: 10, 20, ... %)

Reset answer

```
1+def findavgTime(processes):
2     #TODO: write functions to calculate avgWaitingTime and avgTurnAroundTime of SJF Algorithm (Non-preemptive)
3
4     #! DO NOT CHANGE
5     print("Average waiting time = "+ "{:.2f}".format(avgWaitingTime))
6     print("Average turn around time = "+ "{:.2f}".format(avgTurnAroundTime))
```

Hình 20: SJF Non-preemptive trang quiz



The screenshot shows a Microsoft Edge browser window with a programming assignment. The URL is <http://localhost/question/preview.php?id=72&prevviewid=156&courseld=28&variant=1&correctness=1&marks=2&markdp=2&feedback=1&generalfeedback=1&rightanswer=1&history=0&scrollpos=740>. The page displays a code editor with Python code for SJF Non-preemptive scheduling, followed by a test table and a success message.

Test	Expected	Got
✓ findavgTime([(2,3),(0,4),(4,2),(5,4)])	Average waiting time = 2.00 Average turn around time = 5.25	✓
✓ findavgTime([(3,3),(0,4),(4,2),(5,4),(1,2),(4,5),(2,1),(2,5),(4,5)])	Average waiting time = 8.33 Average turn around time = 11.78	✓
✓ findavgTime([(3,3),(0,4),(4,2),(5,4),(1,2),(4,5),(2,1),(2,5),(4,5)])	Average waiting time = 8.33 Average turn around time = 11.78	✓
✓ findavgTime([(0,5),(2,5),(2,1),(3,5),(3,2),(5,1),(6,2),(8,2)])	Average waiting time = 5.00 Average turn around time = 7.88	✓
✓ findavgTime([(0,2),(2,6),(3,5),(3,2),(5,1),(6,1),(6,2),(7,2),(7,3),(8,2),(7,5),(4,3),(5,6),(10,1)])	Average waiting time = 10.36 Average turn around time = 13.29	✓

Run using the University of Canterbury's jobe server. This is for initial testing only. Please set up your own jobe server as soon as possible. See here.
Passed all tests! ✓
Marks for this submission: 1.00/1.00.

Hình 21: SJF Non-preemptive trả lời đúng

Sau đây là đoạn mã đáp án của câu hỏi SJF Non-preemptive:

```
1 class Process:
2     def __init__(self, arrivalTime, burstTime) -> None:
3         self.arrivalTime = arrivalTime
4         self.burstTime = burstTime
5         self.remainingTime = burstTime
6         self.waitingTime = 0
7         self.turnAroundTime = 0
8         self.completeTime = -1
9
10 def findavgTime(processes):
11     #TODO: write function to calculate avgWaitingTime and avgTurnAroundTime of SJF
12     #Algorithm (Non-preemptive)
13     n = len(processes)
14     lstProcess = []
15     # Tạo các process
16     for process in processes:
17         lstProcess.append(Process(process[0], process[1]))
18
19     # set các thông số cần thiết
20     countProcessComplete = 0
21     runTime = 0
22     #Chạy đến khi nào tất cả các process đều đã hoàn thành
23     while(countProcessComplete < n):
24         index = 0
25         shortestJob = 99999
26         #Tìm process chưa được thực hiện và có burst time nhỏ nhất
27         #Ở trường hợp này burst time cũng là remaining time, vì giải thuật không nhường
28         for i in range(n):
29             if lstProcess[i].arrivalTime <= runTime and lstProcess[i].completeTime == -1:
30                 if lstProcess[i].remainingTime < shortestJob:
31                     #tìm được process thỏa mãn
32                     shortestJob = lstProcess[i].remainingTime
33                     index = i
34
35     #Cập nhật các thông số cho process thỏa mãn
```



```
34     lstProcess [ index ]. waitingTime = runTime - lstProcess [ index ]. arrivalTime
35     runTime += lstProcess [ index ]. remainingTime
36     lstProcess [ index ]. completeTime = runTime
37     lstProcess [ index ]. turnAroundTime = lstProcess [ index ]. waitingTime + lstProcess [
38         index ]. brustTime
39
40     countProcessComplete +=1
41
42     avgWaitingTime = sum([ process .waitingTime for process in lstProcess ]) /n
43     avgTurnAroundTime = sum([ process .turnAroundTime for process in lstProcess ]) /n
44
45     #! DO NOT CHANGE
46     print("Average waiting time = "+ "{:.2f} ".format(avgWaitingTime))
47     print("Average turn around time = "+ "{:.2f} ".format(avgTurnAroundTime))
```

2.3 SJF CPU Scheduling (Preemptive)

Thêm một câu hỏi mới là SJF CPU Scheduling (Preemptive)

The screenshot shows the Moodle quiz editor for a question titled "Programming for SJF CPU Scheduling (preemptive)". The question text is a code snippet for calculating average waiting and turn-around times for processes using the SJF algorithm. It includes instructions to findavgTime and a note about the function's parameters. The editor interface includes fields for category, name, text, mark, feedback, and answer.

Hình 22: SJF Non-preemptive mô tả quiz

Sau đó ta thêm những testcase cần thiết cho quiz này



The screenshot shows a CodeRunner question editor interface. On the left is a sidebar with navigation links like 'Hệ điều hành', 'Participants', 'Badges', 'Competencies', 'Grades', 'General', 'Topic 1' (which is selected), 'Topic 2', 'Topic 3', 'Topic 4', 'Dashboard', 'Site home', 'Calendar', 'Private files', 'Content bank', and 'Site administration'. The main area contains four test cases for the 'findavgTime' function. Each test case includes 'Standard Input' (a list of tuples representing processes), 'Expected output' (the calculated average waiting time and average turn-around time), and 'Test properties' (checkboxes for 'Use as example', 'Display', 'Show', 'Hide rest if fail', 'Mark', and 'Ordering').

Hình 23: SJF Non-preemptive testcase

Chạy thử lại quiz xem đã chính xác chưa

The screenshot shows a preview quiz page for 'Programming for SJF CPU Scheduling (nonpreemptive)'. It displays a question titled 'Question 1' with a note 'Not complete' and 'Marked out of 1.00'. The question text asks to write a function 'findavgTime' for SJF scheduling. Below the question is a 'For example:' section with a table showing three test cases and their results. At the bottom, there is an 'Answer:' field containing the code for the 'findavgTime' function, which is identical to the one shown in the previous screenshot.

Hình 24: SJF Non-preemptive trang quiz



The screenshot shows a programming assignment interface. At the top, it says "in Preview questions Programming for SJF CPU Scheduling (non-preemptive) - Personal - Microsoft Edge". Below that is a code editor window with the following Python code:

```
1 class Process:
2     def __init__(self, arrivalTime, brustTime) -> None:
3         self.arrivalTime = arrivalTime
4         self.brustTime = brustTime
5         self.remainingTime = brustTime
6         self.waitingTime = 0
7         self.turnAroundTime = 0
8         self.completeTime = -1
9
10 + def findavgTime(processes):
11     #TODO: write function to calculate avgWaitingTime and avgTurnAroundTime of SJF Algorithm (Non-preemptive)
12     n = len(processes)
13     lstProcess = []
14     # Tạo các process
15     for process in processes:
16         lstProcess.append(Process(process[0], process[1]))
17
18     # set các thông số cần thiết
19     countProcessComplete = 0
20     runTime = 0
21     #Chạy đến khi nào tất cả các process đều đã hoàn thành
22     #Lặp qua từng thời điểm
23     while(countProcessComplete < n):
24         index = -1
25         shortestJob = 99999
26         #Tìm process chưa được thực hiện và có remaining time nhỏ nhất
27         for i in range(n):
28             if lstProcess[i].arrivalTime <= runTime and lstProcess[i].completeTime == -1:
29                 if lstProcess[i].remainingTime < shortestJob:
30                     #tìm được process thỏa mãn
31                     shortestJob = lstProcess[i].remainingTime
32                     index = i
```

Below the code editor is a "Check" button. Underneath is a table showing test cases and their expected vs. actual results:

Test	Expected	Got
findavgTime([(1,6),(1,8),(2,7),(3,3)])	Average waiting time = 6.75 Average turn around time = 12.75	✓
findavgTime([(2,3),(0,4),(4,2),(5,1)])	Average waiting time = 2.00 Average turn around time = 5.25	✓
findavgTime([(3,3),(0,4),(4,2),(5,4),(1,2),(4,5),(2,1),(2,5),(4,5)])	Average waiting time = 8.00 Average turn around time = 11.44	✓
findavgTime([(0,5),(2,5),(2,1),(3,5),(3,2),(5,1),(6,2),(0,2)])	Average waiting time = 4.25 Average turn around time = 7.12	✓
findavgTime([(0,2),(2,6),(3,5),(3,2),(5,3),(6,1),(6,2),(7,2),(7,3),(6,2),(7,5),(4,3),(5,6),(10,1)])	Average waiting time = 8.71 Average turn around time = 11.64	✓

At the bottom, there is a yellow bar with the text "Run using the University of Canterbury's Jobe server. This is for initial testing only. Please set up your own Jobe server as soon as possible. See here." and a green bar below it saying "Passed all tests! ✓".

Hình 25: SJF Non-preemptive trả lời đúng

Sau đây là đoạn mã đáp án của câu hỏi SJF Non-preemptive:

```
1 class Process:
2     def __init__(self, arrivalTime, brustTime) -> None:
3         self.arrivalTime = arrivalTime
4         self.brustTime = brustTime
5         self.remainingTime = brustTime
6         self.waitingTime = 0
7         self.turnAroundTime = 0
8         self.completeTime = -1
9
10 + def findavgTime(processes):
11     #TODO: write function to calculate avgWaitingTime and avgTurnAroundTime of SJF
12     #Algorithm (Non-preemptive)
13     n = len(processes)
14     lstProcess = []
15     # Tạo các process
16     for process in processes:
17         lstProcess.append(Process(process[0], process[1]))
18
19     # set các thông số cần thiết
20     countProcessComplete = 0
21     runTime = 0
22     #Chạy đến khi nào tất cả các process đều đã hoàn thành
23     #Lặp qua từng thời điểm
24     while(countProcessComplete < n):
25         index = -1
26         shortestJob = 99999
27         #Tìm process chưa được thực hiện và có remaining time nhỏ nhất
28         for i in range(n):
29             if lstProcess[i].arrivalTime <= runTime and lstProcess[i].completeTime == -1:
30                 if lstProcess[i].remainingTime < shortestJob:
31                     #tìm được process thỏa mãn
32                     shortestJob = lstProcess[i].remainingTime
33                     index = i
```



```
35     runTime += 1
36     #nếu không tìm được process thỏa mãn thì skip
37     if index == -1:
38         continue
39     #Cập nhật các thông số cho process thỏa mãn
40     lstProcess[index].remainingTime -= 1
41     if lstProcess[index].remainingTime == 0: #process hoàn thành
42         lstProcess[index].completeTime = runTime
43         lstProcess[index].turnAroundTime = runTime - lstProcess[index].arrivalTime
44         lstProcess[index].waitingTime = lstProcess[index].turnAroundTime -
45         lstProcess[index].burstTime
46         countProcessComplete +=1
47
48     avgWaitingTime = sum([process.waitingTime for process in lstProcess])/n
49     avgTurnAroundTime = sum([process.turnAroundTime for process in lstProcess])/n
50
51     #! DO NOT CHANGE
52     print("Average waiting time = "+ "{:.2f}".format(avgWaitingTime))
53     print("Average turn around time = "+ "{:.2f}".format(avgTurnAroundTime))
54 findavgTime([(1,6),(1,8),(2,7),(3,3)])
```

2.4 LRTF Scheduling

Thêm một câu hỏi mới là LRTF CPU Scheduling

The screenshot shows the Moodle Quiz editor interface. On the left, there's a sidebar with navigation links like 'Hệ điều hành', 'Participants', 'Badges', 'Competencies', 'Grades', 'General', 'Topic 1' (which is selected), 'Topic 2', 'Topic 3', 'Topic 4', 'Dashboard', 'Site home', 'Calendar', 'Private files', 'Content bank', and 'Site administration'. The main area has fields for 'Current category' (set to 'Default for Hệ điều hành (6)') and 'Question name' ('Programming for LRTF CPU Scheduling'). The 'Question text' field contains the provided Python code for the `findavgTime` function. Below the text area, there are fields for 'Default mark' (set to 1) and 'General feedback'. At the bottom, there's an 'ID number' field.

Hình 26: LRTF mô tả quiz

Sau đó ta thêm những testcase cần thiết cho quiz này



Trường đại học Bách khoa - Đại học Quốc gia TP.HCM Khoa Khoa học và Kỹ thuật máy tính

The screenshot shows the 'Editing a CodeRunner question' page in Moodle. On the left, a sidebar lists navigation options like 'Hệ điều hành', 'Participants', 'Badges', 'Competencies', 'Grades', 'General', 'Topic 1' (which is selected), 'Topic 2', 'Topic 3', 'Topic 4', 'Dashboard', 'Site home', 'Calendar', 'Private files', 'Content bank', and 'Site administration'. The main area displays two test cases for the 'findavgTime' function. Test case 4 has input 'findavgTime([(0,5),(2,5),(2,1),(3,5),(3,2),(5,1),(6,2),(8,2)])' and output 'Average waiting time = 13.00\nAverage turn around time = 15.88'. Test case 5 has input 'findavgTime([(0,2),(2,6),(3,5),(3,2),(5,1),(6,1),(6,2),(7,2),(7,3),(6,2),(7,5),(4,3),(5,6),(10,1)])' and output 'Average waiting time = 24.64\nAverage turn around time = 27.57'. Both test cases have a mark of 1.000 and ordering values of 40 and 50 respectively.

Hình 27: LRTF testcase

Chạy thử lại quiz xem đã chính xác chưa

The screenshot shows the 'Preview question Programming for LRTF CPU Scheduling - Personal - Microsoft Edge' page. It displays a question titled 'Question 1' with a note 'Not complete' and 'Marked out of 1.00'. The question text asks to write a function 'findavgTime' to calculate average waiting time and average turn around time for the LRTF algorithm. It provides a template code snippet and notes that the function takes a list of processes as input, where each process is a tuple of arrival time and burst time. The example table shows three test cases with their expected results. Below the table, there is a text area for answers and a note about the penalty regime.

Test	Result
findavgTime([(0,2),(0,3),(2,2),(3,5),(4,4)])	Average waiting time = 9.00 Average turn around time = 12.20
findavgTime([(2,3),(0,4),(4,2),(5,4)])	Average waiting time = 5.50 Average turn around time = 8.75
findavgTime([(3,3),(0,4),(4,2),(5,4),(1,2),(4,5),(2,1),(2,5),(4,5)])	Average waiting time = 20.78 Average turn around time = 24.22

Hình 28: LRTF trang quiz



in Preview questions Programming for LRTF CPU Scheduling - Personal - Microsoft Edge
localhost/question/preview.php?id=4&prevviewid=50;courseId=28;variant=1&correctness=1&marks=2&markdp=2&feedback=1&generalfeedback=1&rightanswer=1&history=0&scrollpos=296

```
Answer: (penalty regime: 10, 20, ... %)
14     # Tạo các process
15     for process in processes:
16         lstProcess.append(Process(process[0], process[1]))
17 
18     # set các thông số cần thiết
19     countProcessComplete = 0
20     runTime = 0
21 
22     #Chạy đến khi nào tất cả các process đều đã hoàn thành
23     #Lặp qua từng thời điểm
24     while(countProcessComplete < n):
25         index = -1
26         longestJob = -99999
27 
28         #Tìm process chưa được thực hiện và có remaining time lớn nhất
29         for i in range(n):
30             if lstProcess[i].arrivalTime <= runTime and lstProcess[i].completeTime == -1:
31                 if lstProcess[i].remainingTime > longestJob:
32                     #tim được process thỏa mãn
33                     longestJob = lstProcess[i].remainingTime
34                     index = i
35 
36         runTime += 1
37 
38         #nếu không tìm được process thỏa mãn thì skip
```

Check

Test	Expected	Got
✓ findavgTime([(0,2),(0,3),(2,2),(3,5),(4,4)])	Average waiting time = 9.00 Average turn around time = 12.20	Average waiting time = 9.00 Average turn around time = 12.20 ✓
✓ findavgTime([(2,3),(0,4),(4,2),(5,4)])	Average waiting time = 5.50 Average turn around time = 8.75	Average waiting time = 5.50 Average turn around time = 8.75 ✓
✓ findavgTime([(3,3),(0,4),(4,2),(5,4),(1,2),(4,5),(2,1),(2,5),(4,5)])	Average waiting time = 20.78 Average turn around time = 24.22	Average waiting time = 20.78 Average turn around time = 24.22 ✓
✓ findavgTime([(0,5),(2,5),(2,1),(3,5),(3,2),(5,1),(6,2),(8,2)])	Average waiting time = 13.00 Average turn around time = 15.88	Average waiting time = 13.00 Average turn around time = 15.88 ✓
✓ findavgTime([(0,2),(2,6),(3,5),(3,2),(5,1),(6,1),(6,2),(7,2),(7,3),(6,2),(7,5),(4,3),(5,6),(10,1)])	Average waiting time = 24.64 Average turn around time = 27.57	Average waiting time = 24.64 Average turn around time = 27.57 ✓

Run using the University of Canterbury's jobe server. This is for initial testing only. Please set up your own jobe server as soon as possible. See here.

Passed all tests! ✓

Marks for this submission: 1.00/1.00.

Hình 29: LRTF trả lời đúng

Sau đây là đoạn mã đáp án của câu hỏi LRTF CPU scheduling:

```
1 class Process:
2     def __init__(self, arrivalTime, brustTime) -> None:
3         self.arrivalTime = arrivalTime
4         self.brustTime = brustTime
5         self.remainingTime = brustTime
6         self.waitingTime = 0
7         self.turnAroundTime = 0
8         self.completeTime = -1
9 
10    def findavgTime(processes):
11        #TODO: write function to calculate avgWaitingTime and avgTurnAroundTime of LRTF
12        #Algorithm
13        n = len(processes)
14        lstProcess = []
15        # Tạo các process
16        for process in processes:
17            lstProcess.append(Process(process[0], process[1]))
18 
19        # set các thông số cần thiết
20        countProcessComplete = 0
21        runTime = 0
22 
23        #Chạy đến khi nào tất cả các process đều đã hoàn thành
24        #Lặp qua từng thời điểm
25        while(countProcessComplete < n):
26            index = -1
27            longestJob = -99999
28 
29            #Tìm process chưa được thực hiện và có remaining time lớn nhất
30            for i in range(n):
31                if lstProcess[i].arrivalTime <= runTime and lstProcess[i].completeTime == -1:
32                    if lstProcess[i].remainingTime > longestJob:
33                        longestJob = lstProcess[i].remainingTime
34                        index = i
35 
36            runTime += 1
37 
38            #nếu không tìm được process thỏa mãn thì skip
```



```
35     runTime += 1
36     #nếu không tìm được process thỏa mãn thì skip
37     if index == -1:
38         continue
39     #Cập nhật các thông số cho process thỏa mãn
40     lstProcess[index].remainingTime -= 1
41     if lstProcess[index].remainingTime == 0: #process hoàn thành
42         lstProcess[index].completeTime = runTime
43         lstProcess[index].turnAroundTime = runTime - lstProcess[index].arrivalTime
44         lstProcess[index].waitingTime = lstProcess[index].turnAroundTime -
45             lstProcess[index].brustTime
46         countProcessComplete +=1
47
48     avgWaitingTime = sum([process.waitingTime for process in lstProcess])/n
49     avgTurnAroundTime = sum([process.turnAroundTime for process in lstProcess])/n
50
51     #! DO NOT CHANGE
52     print("Average waiting time = "+ "{:.2f}".format(avgWaitingTime))
53     print("Average turn around time = "+ "{:.2f}".format(avgTurnAroundTime))
54 findavgTime([(0 ,2) ,(0 ,3) ,(2 ,2) ,(3 ,5) ,(4 ,4) ])
```

2.5 Round Robin Scheduling

Thêm một câu hỏi mới là Round Robin CPU Scheduling

The screenshot shows a Moodle quiz editor interface. On the left, there's a sidebar with navigation links like 'Hệ điều hành', 'Participants', 'Badges', 'Competencies', 'Grades', 'General', 'Topic 1' (which is selected), 'Topic 2', 'Topic 3', 'Topic 4', 'Dashboard', 'Site home', 'Calendar', 'Private files', 'Content bank', and 'Site administration'. The main area has tabs for 'First site Demo', 'Question text', 'Default mark', 'General feedback', and 'Answer'. In the 'Question text' tab, there's a code editor containing Python code for calculating average waiting and turn-around times for processes using the RoundRobin algorithm. Below the code, there's explanatory text and a note about the 'findavgTime' function. The 'Default mark' field is set to 1, and the 'General feedback' field contains a placeholder for feedback.

Hình 30: Round Robin mô tả quiz

Sau đó ta thêm những testcase cần thiết cho quiz này



The screenshot shows a web-based interface for managing test cases. On the left, there's a sidebar with navigation links for 'Hệ điều hành', 'Participants', 'Badges', 'Competencies', 'Grades', 'General', 'Topic 1' (which is selected), 'Topic 2', 'Topic 3', 'Topic 4', 'Dashboard', 'Site home', 'Calendar', 'Private files', 'Content bank', and 'Site administration'. The main area is titled 'test cases' and contains three entries:

- Test case 1:** Standard Input: `findavgTime([(0,3),(0,4),(0,3)],1)`; Expected output: Average waiting time = 5.33, Average turn around time = 8.67; Extra template data: None; Test properties: Use as example checked, Display Show, Hide rest if fail unchecked, Mark 1.000, Ordering 10.
- Test case 2:** Standard Input: `findavgTime([(0,10),(0,5),(0,8)],2)`; Expected output: Average waiting time = 12.00, Average turn around time = 19.67; Extra template data: None; Test properties: Use as example checked, Display Show, Hide rest if fail unchecked, Mark 1.000, Ordering 20.
- Test case 3:** Standard Input: `findavgTime([(1,6),(1,8),(2,7),(3,3)],2)`; Expected output: None; Extra template data: None; Test properties: Use as example checked, Display Show, Hide rest if fail unchecked, Mark 1.000, Ordering 20.

Hình 31: Round Robin testcase

Chạy thử lại quiz xem đã chính xác chưa

The screenshot shows a web-based quiz page for 'Programming for RoundRobin CPU Scheduling'. At the top, it says 'Preview question Programming for RoundRobin CPU Scheduling - Personal - Microsoft Edge' and the URL 'localhost/question/preview.php?id=25&coursed=2'. The page has a light blue header with the question number 'Question 1' and a note 'Not complete'.

Question 1: Not complete
Marked out of 1.00

Viết function findavgTime để xuất ra thời gian đợi trung bình và thời gian xoay vòng trung bình của giải thuật RoundRobin CPU Scheduling theo template như sau:

```
# TODO: write function to calculate avgWaitingTime and avgTurnAroundTime of RoundRobin Algorithm
# DO NOT CHANGE
print("Average waiting time = "+ "{:.2f}".format(avgWaitingTime))
print("Average turn around time = "+ "{:.2f}".format(avgTurnAroundTime))
```

Hàm findavgTime được truyền vào là một list của các tuple 2 phần tử, phần tử thứ nhất sẽ là arrival time của process, phần tử thứ hai sẽ là burst time của process và quantumTime là số nguyên. Đề đúng với kết quả, in đúng theo kết quả testcase dưới đây, lấy 2 chữ số sau chữ số thập phân (như code mẫu đã sinh ra).

For example:

Test	Result
<code>findavgTime([(0,3),(0,4),(0,3)],1)</code>	Average waiting time = 5.33 Average turn around time = 8.67
<code>findavgTime([(0,10),(0,5),(0,8)],2)</code>	Average waiting time = 12.00 Average turn around time = 19.67

Answer: (penalty regime: 10, 20, ... %)

Reset answer

```
1. def findavgTime(processes, quantumTime):
2.     # TODO: write function to calculate avgWaitingTime and avgTurnAroundTime of RoundRobin Algorithm
3.
4.     #! DO NOT CHANGE
5.     print("Average waiting time = "+ "{:.2f}".format(avgWaitingTime))
6.     print("Average turn around time = "+ "{:.2f}".format(avgTurnAroundTime))
7.
```

Hình 32: Round Robin trang quiz



The screenshot shows a Microsoft Edge browser window with a programming assignment. The title bar reads "in Preview question: Programming for RoundRobin CPU Scheduling - Personal - Microsoft Edge". The URL is "localhost/question/preview.php?id=55&prevviewid=20&courseid=18&variant=1&correctness=1&marks=2&markdp=2&feedback=1&generalfeedback=1&rightanswer=1&history=0&scrollipos=592".
Answer: (penalty regime 10, 20, ... %)
Reset answer
1 class Process:
2 def __init__(self, arrivalTime, burstTime) -> None:
3 self.arrivalTime = arrivalTime
4 self.burstTime = burstTime
5 self.remainingTime = burstTime
6 self.waitingTime = 0
7 self.turnAroundTime = 0
8 self.completeTime = -1
9
10 def findavgTime(processes, quantumTime):
11 #TODO: write function to calculate avgWaitingTime and avgTurnAroundTime of RoundRobin Algorithm
12 n = len(processes)
13 lstProcess = []
14 # Tao các process
15 for process in processes:
16 lstProcess.append(Process(process[0], process[1]))
17
18 # set các thông số cần thiết
19 countProcessComplete = 0
20 runTime = 0
21 index = 0
22
23
Check

Test	Expected	Got
✓ findavgTime([(0,3),(0,4),(0,3)],1)	Average waiting time = 5.33 Average turn around time = 8.67	Average waiting time = 5.33 Average turn around time = 8.67 ✓
✓ findavgTime([(0,10),(0,5),(0,8)],2)	Average waiting time = 12.00 Average turn around time = 19.67	Average waiting time = 12.00 Average turn around time = 19.67 ✓
✓ findavgTime([(1,6),(1,8),(2,7),(3,3)],2)	Average waiting time = 12.75 Average turn around time = 18.75	Average waiting time = 12.75 Average turn around time = 18.75 ✓
✓ findavgTime([(0,5),(2,5),(2,1),(3,5),(3,2),(5,1),(6,2),(8,2)],3)	Average waiting time = 13.62 Average turn around time = 16.50	Average waiting time = 13.62 Average turn around time = 16.50 ✓
✓ findavgTime([(0,2),(2,6),(3,5),(3,2),(5,1),(6,1),(6,2),(7,2),(7,3),(8,2),(7,5),(4,3),(5,6),(10,1)],4)	Average waiting time = 27.21 Average turn around time = 30.14	Average waiting time = 27.21 Average turn around time = 30.14 ✓

Run using the University of Canterbury's jobe server. This is for initial testing only. Please set up your own jobe server as soon as possible. See here.
Passed all tests! ✓

Marks for this submission: 1.00/1.00.

Hình 33: Round Robin trả lời đúng

Sau đây là đoạn mã đáp án của câu hỏi Round Robin CPU scheduling:

```
1 class Process:  
2     def __init__(self, arrivalTime, burstTime) -> None:  
3         self.arrivalTime = arrivalTime  
4         self.burstTime = burstTime  
5         self.remainingTime = burstTime  
6         self.waitingTime = 0  
7         self.turnAroundTime = 0  
8         self.completeTime = -1  
9  
10    def findavgTime(processes, quantumTime):  
11        #TODO: write function to calculate avgWaitingTime and avgTurnAroundTime of  
12        #RoundRobin Algorithm  
13        n = len(processes)  
14        lstProcess = []  
15        # Tao các process  
16        for process in processes:  
17            lstProcess.append(Process(process[0], process[1]))  
18  
19        # set các thông số cần thiết  
20        countProcessComplete = 0  
21        runTime = 0  
22        index = 0  
23        quantumCount = 0  
24        #Chạy đến khi nào tất cả các process đều đã hoàn thành  
25        #Lặp qua từng thời điểm  
26        while(countProcessComplete < n):  
27            if quantumCount == 0:  
28                #Tìm process chưa được thực hiện và có remaining time nhỏ nhất  
29                for i in range(index, index + n):  
30                    if lstProcess[i%n].arrivalTime <= runTime and lstProcess[i%n].  
31                    completeTime == -1:  
32                        #tìm được process thõa mãn  
33                        index = i%n  
34                        quantumCount = quantumTime  
35                        runTime += 1
```



```
35     quantumCount == 1
36     #Cập nhật các thông số cho process thỏa mãn
37     lstProcess [index].remainingTime == 1
38     if lstProcess [index].remainingTime == 0: #process hoàn thành
39         lstProcess [index].completeTime = runTime
40         lstProcess [index].turnAroundTime = runTime - lstProcess [index].arrivalTime
41         lstProcess [index].waitingTime = lstProcess [index].turnAroundTime -
42             lstProcess [index].brustTime
43         countProcessComplete +=1
44         print(index+1, lstProcess [index].completeTime, lstProcess [index].-
45             waitingTime, lstProcess [index].turnAroundTime)
46     avgWaitingTime = sum([process.waitingTime for process in lstProcess])/n
47     avgTurnAroundTime = sum([process.turnAroundTime for process in lstProcess])/n
48
49     #! DO NOT CHANGE
50     print("Average waiting time = "+ "{:.2f}".format(avgWaitingTime))
51     print("Average turn around time = "+ "{:.2f}".format(avgTurnAroundTime))
52
53     findavgTime([(0,3),(0,4),(0,3)],1)
54     findavgTime([(0,10),(0,5),(0,8)],2)
```



3 Xây dựng môi trường và build testcase cho DemandPaging - Page Replacement

Dối với các giải thuật thay trang, em sẽ tạo các câu quiz yêu cầu tính số lần thay trang.

3.1 FIFO

Thêm một câu hỏi mới là FIFO page replacement

The screenshot shows the Moodle Question editor interface. On the left, there is a sidebar with navigation links like 'Hệ điều hành', 'Participants', 'Badges', 'Competencies', 'Grades', 'General', 'Topic 1' (which is selected), 'Topic 2', 'Topic 3', 'Topic 4', 'Dashboard', 'Site home', 'Calendar', 'Private files', 'Content bank', 'Site administration', and 'Add a block'. The main area is titled 'Editing a CodeRunner question' and shows 'Question type details'. Under the 'General' tab, the 'Question name' is 'FIFO page replacement' and the 'Question text' contains the following code and instructions:

```
Viết function findNumOfPageReplacement để in ra màn hình số lần thay trang, với giá trị truyền vào là dãy những trang tham chiếu và số frames với giải thuật FIFO
def findnumOfPageReplacement(pageReference, numOfframes):
    pass
```

Để đúng với kết quả, in đúng theo kết quả testcase dưới đây, chỉ in đúng số trang được thay thế.

Below the text area, there is a note: '(i) A draft version of this text was automatically restored.'

Under 'Default mark', the value is '1'.

Hình 34: FIFO mô tả quiz

Sau đó ta thêm những testcase cần thiết cho quiz này



The screenshot shows the Moodle CodeRunner question editor. On the left, there's a sidebar with navigation links like 'Hệ điều hành', 'Participants', 'Badges', 'Competencies', 'Grades', 'General', 'Topic 1' (which is selected), 'Topic 2', 'Topic 3', 'Topic 4', 'Dashboard', 'Site home', 'Calendar', 'Private files', 'Content bank', 'Site administration', and 'Add a block'. The main area is titled 'Editing a CodeRunner question' and shows 'Test cases'. It contains two test cases, each with 'Standard Input', 'Expected output', and 'Extra template data' sections. Below each test case are 'Test properties' with checkboxes for 'Use as example', 'Display', 'Show', 'Hide rest if fail', 'Mark', and 'Ordering'. Test case 1 has input '[7,0,1,2,0,3,0,4,2,3,0,3,0,3,2,1,2,0,1,7,0,1],3', expected output '15', and ordering 10. Test case 2 has input '[2,5,10,1,2,2,6,9,1,2,10,2,6,1,2,1,6,9,5,1],3', expected output '16', and ordering 20.

Hình 35: FIFO testcase

Chạy thử lại quiz xem đã chính xác chưa

The screenshot shows the Moodle preview page for a question. The top bar says 'Preview question FIFO page replacement - Personal - Microsoft Edge'. The question title is 'Question 1' (Not complete, Marked out of 1.00). The question text asks to write a function 'findNumOfPageReplacement' that takes a list of page references and a number of frames, and returns the number of page replacements. It provides an example table:

Test	Result
findNumOfPageReplacement([7,0,1,2,0,3,0,4,2,3,0,3,0,3,2,1,2,0,1,7,0,1],3)	15
findNumOfPageReplacement([2,5,10,1,2,2,6,9,1,2,10,2,6,1,2,1,6,9,5,1],3)	16
findNumOfPageReplacement([2,5,10,1,2,2,6,9,1,2,10,2,6,1,2,1,6,9,5,1],4)	12

The answer field shows the code: 'def findNumOfPageReplacement(pageReference, numOfFrames): pass'. There are 'Reset answer' and 'Check' buttons at the bottom.

Hình 36: FIFO trang quiz



The screenshot shows a programming assignment interface. At the top, it says "in Preview question FIFO page replacement - Personal - Microsoft Edge". Below that is the URL "localhost/question/preview.php?unit=228;courseid=28;variant=1&correctness=1&marks=2&markdp=2&feedback=1&generalfeedback=1&rightanswer=1&history=0&scrollpos=296". The interface has two main sections: "Test" and "Result". The "Test" section contains three rows of test cases:

Test	Result
findNumOfPageReplacement([7,0,1,2,0,3,0,4,2,3,0,3,0,3,2,1,2,0,1,7,0,1],3)	15
findNumOfPageReplacement([2,5,10,1,2,1,2,0,6,9,1,2,10,2,6,1,1,2,1,6,9,5,1],3)	16
findNumOfPageReplacement([2,5,10,1,2,1,2,0,6,9,1,2,10,2,6,1,2,1,6,9,5,1],4)	12

Below the test section is a "Reset answer" button and a "Check" button. The "Result" section shows the output of the provided Python code for each test case, with checkmarks indicating they are correct. The code itself is:

```
1 def findNumOfPageReplacement(pageReference, numOfframes):
2     frames = [-1]*numOfframes
3     currentIndex = 0
4     numOfPageReplacement = 0
5     for page in pageReference:
6         if page not in frames:
7             frames[currentIndex%numOfframes] = page
8             numOfPageReplacement +=1
9             currentIndex +=1
10    print(numOfPageReplacement)
```

At the bottom of the interface, there is a yellow bar with the text "Run using the University of Canterbury's Jobe server. This is for initial testing only. Please set up your own Jobe server as soon as possible. See here."

Hình 37: FIFO trả lời đúng

Sau đây là đoạn mã đáp án của câu hỏi FIFO replacement:

```
1 def findNumOfPageReplacement (pageReference , numOfframes) :
2     frames = [-1]*numOfframes
3     currentIndex = 0
4     numOfPageReplacement = 0
5     for page in pageReference:
6         if page not in frames:
7             frames [currentIndex%numOfframes] = page
8             numOfPageReplacement +=1
9             currentIndex +=1
10    print (numOfPageReplacement)
```

3.2 Optimal

Thêm một câu hỏi mới là Optimal page replacement



The screenshot shows the Moodle Quiz editor interface. On the left, a sidebar menu includes 'Hệ điều hành', 'Participants', 'Badges', 'Competencies', 'Grades', 'General', and 'Topic 1' (which is selected). The main area displays 'Question type details' for 'FIFO page replacement'. It has sections for 'Current category' (Default for Hệ điều hành (10)) and 'Save in category' (Default for Hệ điều hành (10)). The 'Question name' is 'FIFO page replacement'. The 'Question text' contains the following code and note:

```
Viết function findNumOfPageReplacement để in ra màn hình số lần thay trang, với giá trị truyền vào là dãy những trang tham chiếu và số frames với giải thuật FIFO
def findNumOfPageReplacement(pageReference, numOfFrames):
    pass
Để đúng với kết quả, in đúng theo kết quả testcase dưới đây, chỉ in đúng số trang được thay thế.
```

A note below states: 'A draft version of this text was automatically restored.'

Below the question text, there are fields for 'Default mark' (set to 1) and 'General feedback'.

Hình 38: FIFO mô tả quiz

Sau đó ta thêm những testcase cần thiết cho quiz này

The screenshot shows the Moodle Quiz editor interface with 'Test cases' selected in the sidebar. The 'Topic 1' section is still active. It lists two test cases:

Test case	Standard Input	Expected output	Extra template data	Test properties
Test case 1	findNumberOfPageReplacement([7,0,1,2,0,3,0,4,2,3,0,3,0,3,2,1,2,0,1,7,0,1],3)	15		<input checked="" type="checkbox"/> Use as example <input type="checkbox"/> Display Show <input type="checkbox"/> Hide rest if fail <input type="checkbox"/> Mark 1.000 Ordering 10
Test case 2	findNumberOfPageReplacement([2,5,10,1,2,2,6,9,1,2,10,2,6,1,2,1,6,9,5,1],3)	16		<input checked="" type="checkbox"/> Use as example <input type="checkbox"/> Display Show <input type="checkbox"/> Hide rest if fail <input type="checkbox"/> Mark 1.000 Ordering 20

Hình 39: FIFO testcase

Chạy thử lại quiz xem đã chính xác chưa



```
7     frames[currentIndex%numOfFrames] = page
8     numOfPageReplacement +=1
9     currentIndex +=1
10
11     print(numOfPageReplacement)
```

3.3 LRU

Thêm một câu hỏi mới là LRU page replacement

The screenshot shows the Moodle quiz editor interface. On the left, there's a sidebar with navigation links like 'Hệ điều hành', 'Participants', 'Badges', 'Competencies' (which is checked), 'Grades', 'General', 'Topic 1' (which is selected), 'Topic 2', 'Topic 3', 'Topic 4', 'Dashboard', 'Site home', 'Calendar', 'Private files', 'Content bank', 'Site administration', and 'Add a block'. The main area has sections for 'Current category' (set to 'Default for Hệ điều hành (10)') and 'Save in category' (set to 'Default for Hệ điều hành (10)'). The 'Question name' field contains 'LRU page replacement'. The 'Question text' field contains the following code and instructions:

```
Viết function findNumOfPageReplacement để in ra màn hình số lần thay trang, với giá trị truyền vào là dãy những trang tham chiếu và số frames với giải thuật LRU
def findNumOfPageReplacement(pageReference, numOfFrames):
    pass
```

Để dùng với kết quả, in đúng theo kết quả testcase dưới đây, chỉ in đúng số trang được thay thế.

Hình 42: LRU mô tả quiz

Sau đó ta thêm những testcase cần thiết cho quiz này



The screenshot shows a web-based interface for editing a CodeRunner question. On the left, a sidebar lists various site sections like 'Hệ điều hành', 'Participants', 'Badges', 'Competencies', 'Grades', 'General', and several 'Topic' sections. 'Topic 1' is currently selected. The main area is titled 'test cases' and contains three entries:

- Test case 1:** Standard Input: `findNumOfPageReplacement ([7,0,1,2,0,3,0,4,2,3,0,3,0,3,2,1,2,0,1,7,0,1],3)`; Expected output: `12`.
Test properties: Use as example, Display Show, Hide rest if fail, Mark 1.000, Ordering 10.
- Test case 2:** Standard Input: `findNumOfPageReplacement ([2,5,10,1,2,2,6,9,1,2,10,2,6,1,2,1,6,9,5,1],3)`; Expected output: `15`.
Test properties: Use as example, Display Show, Hide rest if fail, Mark 1.000, Ordering 20.
- Test case 3:** Standard Input: `findNumOfPageReplacement ([2,5,10,1,2,2,6,9,1,2,10,2,6,1,2,1,6,9,5,1],4)`.

Hình 43: LRU testcase

Chạy thử lại quiz xem đã chính xác chưa

The screenshot shows a 'Preview question' page for a quiz. At the top, it says 'Question 1 Not complete Marked out of 1.00'. The main content area contains the following text:

Viết function `findNumOfPageReplacement` để ra màn hình số lần thay trang, với giá trị truyền vào là dãy những trang tham chiếu và số frames với giải thuật LRU

```
def findNumOfPageReplacement(pageReference, numOfFrames):
    pass
```

Để đúng với kết quả, in đúng theo kết quả testcase dưới đây, chỉ in đúng số trang được thay thế.

For example:

Test	Result
<code>findNumOfPageReplacement([7,0,1,2,0,3,0,4,2,3,0,3,0,3,2,1,2,0,1,7,0,1],3)</code>	12
<code>findNumOfPageReplacement([2,5,10,1,2,2,6,9,1,2,10,2,6,1,2,1,6,9,5,1],3)</code>	15
<code>findNumOfPageReplacement([2,5,10,1,2,2,6,9,1,2,10,2,6,1,2,1,6,9,5,1],4)</code>	10

Answer: (penalty regime: 10, 20, ... %)

Reset answer

```
1 def findNumOfPageReplacement(pageReference, numOfFrames):
2     pass
```

Check

Hình 44: LRU trang quiz



The screenshot shows a Microsoft Edge browser window with a Python code editor and a test results section.

Answer: (penalty regime: 10, 20, ... %)

Reset answer

```
1 def findNumOfPageReplacement(pageReference, numOfFrames):
2     frames = [-1]*numOfFrames
3     numOfPageReplacement = 0
4     for i in range(len(pageReference)):
5         if pageReference[i] not in frames:
6             index = 0
7             maxLength = -99999
8             for j in range(numOfFrames):
9                 if frames[j] not in pageReference[:i]:
10                     index = j
11                     break
12                 if pageReference[:i][::-1].index(frames[j]) > maxLength:
13                     maxLength = pageReference[:i][::-1].index(frames[j])
14                     index = j
15             frames[index] = pageReference[i]
16             numOfPageReplacement += 1
17     print(numOfPageReplacement)
```

Check

Test

- ✓ findNumOfPageReplacement([7,0,1,2,0,3,0,4,2,3,0,3,0,3,2,1,2,0,1,7,0,1],3)
- ✓ findNumOfPageReplacement([2,5,10,1,2,2,6,9,1,2,10,2,6,1,2,1,6,9,5,1],3)
- ✓ findNumOfPageReplacement([2,5,10,1,2,2,6,9,1,2,10,2,6,1,2,1,6,9,5,1,2])
- ✓ findNumOfPageReplacement([2,5,10,1,2,1,2,2,6,9,1,2,2,6,9,1,2,10,2,6,1,2,1,6,9,5,1,2])
- ✓ findNumOfPageReplacement([2,5,10,1,2,1,2,2,6,9,1,2,2,6,9,1,2,10,2,6,1,2,1,6,9,5,1,2])

Run using the University of Canterbury's job server. This is for initial testing only. Please set up your own job server as soon as possible. See here.

Passed all tests! ✓

Marks for this submission: 1.00/1.00.

Hình 45: LRU trả lời đúng

Sau đây là đoạn mã đáp án của câu hỏi LRU replacement:

```
1 def findNumOfPageReplacement (pageReference , numOffFrames) :
2     frames = [-1]*numOffFrames
3     numOffPageReplacement = 0
4     for i in range (len (pageReference)) :
5         if pageReference [i] not in frames :
6             index = 0
7             maxLength = -99999
8             for j in range (numOffFrames) :
9                 if frames [j] not in pageReference [:i] :
10                     index = j
11                     break
12                 if pageReference [:i][::-1].index (frames [j]) > maxLength :
13                     maxLength = pageReference [:i][::-1].index (frames [j])
14                     index = j
15             frames [index] = pageReference [i]
16             numOffPageReplacement += 1
17     print (numOffPageReplacement)
```

3.4 Second change - Clock

Thêm một câu hỏi mới là Second change - Clock page replacement



The screenshot shows the 'Editing a CodeRunner question' page in Moodle. The left sidebar is titled 'First site Demo' and includes links for 'Hệ điều hành', 'Participants', 'Badges', 'Competencies', 'Grades', 'General', 'Topic 1' (which is selected), 'Topic 2', 'Topic 3', 'Topic 4', 'Dashboard', 'Site home', 'Calendar', 'Private files', 'Content bank', 'Site administration', and 'Add a block'. The main area is titled 'General' and contains fields for 'Current category' (set to 'Default for Hệ điều hành (10)'), 'Save in category' (set to 'Default for Hệ điều hành (10)'), 'Question name' ('Second Change (Clock) page replacement'), 'Question text' (containing code and notes about finding the number of page replacements), 'Default mark' (set to 1), 'General feedback' (empty), and 'ID number' (empty). There are also rich text editors for question text and general feedback.

Hình 46: Clock mô tả quiz

Sau đó ta thêm những testcase cần thiết cho quiz này

The screenshot shows the 'Editing a CodeRunner question' page in Moodle, focusing on the 'Test cases' section. The left sidebar is identical to the previous screenshot. The main area is titled 'Test cases' and lists two test cases. Test case 1 has 'Standard Input' containing 'findNumberOfPageReplacement ([7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1], 3)' and 'Expected output' containing '11'. Test case 2 has 'Standard Input' containing 'findNumberOfPageReplacement ([2, 5, 10, 1, 2, 2, 6, 9, 1, 2, 10, 2, 6, 1, 2, 1, 6, 9, 5, 1], 3)' and 'Expected output' containing '13'. Both test cases have 'Test properties' sections with checkboxes for 'Use as example', 'Display' (set to 'Show'), 'Hide rest if fail', 'Mark' (set to 1.000 or 2.00), and 'Ordering' (set to 10 or 20).

Hình 47: Clock testcase

Chạy thử lại quiz xem đã chính xác chưa



in Preview questions Second Change (Clock) page replacement - Personal - Microsoft Edge
localhost/question/preview.php?id=298&previwick=25&courseid=2&variant=1&correctness=1&marks=2&markdp=2&feedback=1&generalfeedback=1&rightanswer=1&history=0&scrollpos=592

Question 1
Not complete
Marked out of 1.00

Viết function findNumOfPageReplacement để in ra màn hình số lần thay trang, với giá trị truyền vào là dãy những trang tham chiếu và số frames với giải thuật Second change (Clock)

```
def findNumOfPageReplacement(pageReference, numFrames):
    pass
```

Để đúng với kết quả, in đúng theo kết quả testcase dưới đây, chỉ in đúng số trang được thay thế.

For example:

Test	Result
findNumOfPageReplacement([7,0,1,2,0,3,0,4,2,3,0,3,0,3,2,1,2,0,1,7,0,1],3)	11
findNumOfPageReplacement([2,5,10,1,2,2,6,9,1,2,10,2,6,1,2,1,6,9,5,1],3)	13
findNumOfPageReplacement([2,5,10,1,2,2,6,9,1,2,10,2,6,1,2,1,6,9,5,1],4)	11

Answer: (penalty regime: 10, 20, ... %)

Reset answer

```
1 def findNumOfPageReplacement(pageReference, numFrames):
2     pass
```

Check

Hình 48: Clock trang quiz

in Preview questions Second Change (Clock) page replacement - Personal - Microsoft Edge
localhost/question/preview.php?id=298&previwick=25&courseid=2&variant=1&correctness=1&marks=2&markdp=2&feedback=1&generalfeedback=1&rightanswer=1&history=0&scrollpos=592

Answer: (penalty regime: 10, 20, ... %)

Reset answer

```
1 def findNumOfPageReplacement(pageReference, numFrames):
2     frames = [-1]*numFrames
3     secondChange = [False]*numFrames
4     currentIndex = 0
5     numOfPageReplacement = 0
6     for i in range(len(pageReference)):
7         if pageReference[i] in frames:
8             continue
9         elif pageReference[i] not in frames:
10            for j in range(currentIndex, currentIndex+numFrames*2):
11                if secondChange[j] == True:
12                    secondChange[j] = False
13                    continue
14                else:
15                    frames[j] = pageReference[i]
16                    currentIndex = j
17                    numOfPageReplacement += 1
18
19 print(numOfPageReplacement)
```

Check

Test

✓ findNumOfPageReplacement([7,0,1,2,0,3,0,4,2,3,0,3,0,3,2,1,2,0,1,7,0,1],3)
✓ findNumOfPageReplacement([2,5,10,1,2,2,6,9,1,2,10,2,6,1,2,1,6,9,5,1],3)
✓ findNumOfPageReplacement([2,5,10,1,2,2,6,9,1,2,10,2,6,1,2,1,6,9,5,1],4)
✓ findNumOfPageReplacement([2,5,10,1,2,2,6,9,1,2,10,2,6,1,2,1,6,9,5,1],3)
✓ findNumOfPageReplacement([2,5,10,1,2,2,6,9,1,2,10,2,6,1,2,1,6,9,5,1],4)

Run using the University of Canterbury's Jobe server. This is for initial testing only. Please set up your own Jobe server as soon as possible. See here.

Passed all tests! ✓

Marks for this submission: 1.00/1.00.

Hình 49: Clock trả lời đúng

Sau đây là đoạn mã đáp án của câu hỏi Second change - Clock replacement:

```
1 def findNumOfPageReplacement( pageReference , numOfFrames ) :
2     frames = [-1]*numOfFrames
3     secondChange = [ False ]*numOfFrames
4     currentIndex = 0
5     numOfPageReplacement = 0
6     for i in range( len( pageReference ) ) :
```



```
7     if pageReference[ i ] in frames:
8         secondChange[ frames.index( pageReference[ i ]) ] = True
9     elif pageReference[ i ] not in frames:
10        for j in range( currentIndex , currentIndex+numOfFrames*2):
11            if secondChange[ j%numOfFrames ] == True:
12                secondChange[ j%numOfFrames ] = False
13                continue
14            else:
15                frames[ j%numOfFrames ] = pageReference[ i ]
16                currentIndex = j%numOfFrames + 1
17                break
18        numOfPageReplacement += 1
19    print( numOfPageReplacement )
```



4 Đánh giá tổng hợp kết quả đạt được

Như vậy qua bài tập lớn này, em đã xây dựng được môi trường ảo chạy trên CodeRunner hỗ trợ các bạn làm CPU scheduling và Demand Paging. Cụ thể là ở CPU scheduling có các giải thuật cơ bản là: First-Come-First-Serve (FCFS), Shortest-Job-First (SJF) preemptive và non-preemptive, Longest-Remaining-Time-First (LRTF) và Round Robin. Ở Demand Paging là các giải thuật thay trang: First-In-First-Out (FIFO), Optimal, Least Recently Used (LRU), Second-Chance (Clock). Các câu quiz tương đối đơn giản, chủ yếu đánh vào phần giải thuật. Các testcase mang tính chất giới thiệu sẽ được show ra cho các bạn xem, các testcase tricky sẽ được ẩn đi tránh trường hợp các bạn không bao quát được tất cả trường hợp. Nhìn chung các câu quiz đủ để đánh giá chính xác được giải thuật mà các bạn lập trình. Qua bài tập lớn này, em cũng đã hiểu hơn về cách sử dụng Moodle, cũng như module CodeRunner. Ngoài ra, em cũng rèn luyện thêm được các giải thuật CPU Scheduling và Demand Paging nói trên.



Tài liệu tham khảo

- [1] [Moodle](#)
- [2] Abraham Silberschatz, Peter Baer Galvin, Greg Gagne (2017). Operating System Concepts - Tenth Edition