|  |  |
| --- | --- |
| **Name:** Lê Minh Nguyệt  **ID:** 21521211 | **Name:** Trần Trung Tín  **ID:** 21522679 |
| **Name:** Hồ Đức Trưởng  **ID:** 21522730 | **Name:** Nguyễn Công Nguyên  **ID:** 21521200 |
| **Class:** IT007.N11 | **Group Name:** Team |

**OPERATING SYSTEM  
LAB 06’S REPORT**

**SUMMARY**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Task** | | | **Status** | **Page** |
| Section 6.4 | Ex 1. Giải thuật FIFO | * 1. Vẽ lưu đồ giải thuật | Done | 6 |
| * 1. Trình bày tính đúng đắn của lưu đồ bằng cách chạy tay ít nhất 01 test case tối thiểu 10 phần tử | 7 |
| * 1. Thực hiện code cho giải thuật | 10 |
| * 1. Trình bày tính đúng đắn của code bằng cách chạy ít nhất 03 test case | 12 |
| Ex 2. Giải thuật OPT | 2.1. Vẽ lưu đồ giải thuật | Done | 15 |
| 2.2. Trình bày tính đúng đắn của lưu đồ bằng cách chạy tay ít nhất 01 test case tối thiểu 10 phần tử | 18 |
| 2.3. Thực hiện code cho giải thuật | 22 |
| 2.4. Trình bày tính đúng đắn của code bằng cách chạy ít nhất 03 test case | 24 |
| Ex 3. Giải thuật LRU | 3.1. Vẽ lưu đồ giải thuật | Done | 27 |
| 3.2. Trình bày tính đúng đắn của lưu đồ bằng cách chạy tay ít nhất 01 test case tối thiểu 10 phần tử | 30 |
| 3.3. Thực hiện code cho giải thuật | 34 |
| 3.4. Trình bày tính đúng đắn của code bằng cách chạy ít nhất 03 test case | 36 |
| Section 6.5 | Ex 1. Nghịch lý Belady | * 1. Khái niệm nghịch lý Belady | Done | 40 |
| * 1. Sử dụng chương trình đã viết chứng minh nghịch lý Belady |
| Ex 2. Mức độ hiệu quả và tính khả thi của 3 giải thuật | 2.1. Giải thuật bất khả thi nhất | Done | 41 |
| 2.2. Giải thuật phức tạp nhất |

```````````````````````````

**Self-scrores: 9,5**

**Section 6.4**

***\*Thực hiện Menu theo yêu cầu Section 6.4:***

* Tiến hành khai báo chuỗi tham chiếu mặc định def\_seq gồm 11 phần tử (8 chữ số mã số sinh viên và 2 chữ số 0, 0, 7).
* Mảng seq chứa chuỗi tham chiếu muốn áp dụng giải thuật, n là số phần tử của chuỗi.
* Khai báo biến frame\_num chứa số khung trang, do người dùng nhập vào.
* Tạo menu lựa chọn chuỗi tham chiếu: 1 để chọn chuỗi mặc định, 2 để tự nhập chuỗi vào từ bàn phím.
* Dùng vòng lặp do…while… yêu cầu người dùng nhập lại cho đến khi thỏa mãn một trong hai lựa chọn 1 hoặc 2.

Text

Description automatically generated

*Hình 1. Tạo chuỗi tham chiếu mặc định và menu lựa chọn chuỗi tham chiếu*

* Nếu người dùng chọn 2 (choice\_seq==2), cho phép người dùng nhập chuỗi tham chiếu từ bàn phím. Cụ thể số lượng phần tử lưu vào biến n, các giá trị trong chuỗi lưu vào mảng seq[].
* Nếu người dùng chọn chuỗi mặc định, lấy chuỗi trong mảng def\_seq[] lưu vào seq[] và n=11.
* Yêu cầu người dùng nhập số khung trang, lưu vào biến frame\_num.
* Tạo menu lựa chọn một trong ba giải thuật FIFO, OPT, LRU.

Text

Description automatically generated

*Hình 2. Nhập chuỗi tham chiếu và số lượng khung trang từ bàn phím và tạo menu lựa chọn giải thuật*

* Tương tự với lựa chọn chuỗi bên trên, ta dùng vòng lặp do…while… yêu cầu người dùng nhập lại cho đến khi thỏa mãn một trong ba lựa chọn 1, 2 hoặc 3 (choice\_algo nhận các giá trị trong khoảng [1, 3]).

Text

Description automatically generated

*Hình 3. Lựa chọn một trong ba giải thuật và thực thi*

* Nếu người dùng chọn 1, ta thực hiện giải thuật FIFO bằng cách gọi hàm FIFO() với các tham số truyền vào là chuỗi tham chiếu seq, số lượng phần tử n, và số lượng khung trang frame\_num.
* Tương tự với lựa chọn 2 (giải thuật OPT) và 3 (giải thuật LRU).

Text

Description automatically generated

*Hình 4. Một số chú thích trong lưu đồ và code của ba giải thuật*

* Biến available đánh dấu trang có sẵn trong khung hay không, nhận giá trị 0 (không có sẵn, phải thay trang) hoặc 1 (có sẵn, không cần thay).
* Biến replace\_i lưu vị trí khung trang sẽ bị thay ra, nhận giá trị trong khoảng [0, frame\_num-1].
* Biến fault\_num đếm số lượng lỗi trang.
* Mảng faults[n] đánh dấu mỗi trang có xảy ra lỗi hay không, mỗi phần tử ứng với mỗi trang, nhận giá trị 0 (không lỗi) hoặc 1 (có lỗi).
* Bảng total\_frames[frame\_num][n]:

Chuỗi tham chiếu seq[j]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Khung trang thứ i | j=0 | j=1 | … | j=n-1 |
| i=0 |  |  |  |  |
| i=1 |  |  |  |  |
| … |  |  |  |  |
| i=frame\_num-1 |  |  |  |  |

***total\_frames[i][j]***

***Ex 1.Giải thuật FIFO***

* 1. **Vẽ lưu đồ giải thuật**

Diagram

Description automatically generated

Diagram

Description automatically generated

*Hình 5. Lưu đồ giải thuật FIFO*

* Khởi tạo biến fault\_num=0 và tất cả giá trị trong mảng faults[]=0, do ban đầu chưa có lỗi trang nào.
* replace\_i khởi tạo=0: vị trí khung trang đầu tiên thêm vào.
* Ta thực hiện vòng lặp bắt đầu j=0 cho đến n-1 (j đi qua từng phần tử của chuỗi tham chiếu). Tại mỗi phần tử:

+ Tiến hành tìm trang tại vị trí j, xem có trong các khung trang hiện tại không. Nếu có, không thực hiện thay trang. Ngược lại thực hiện thay trang vào vị trí replace\_i, sau đó tăng replace\_i lên 1 đơn vị nhưng chỉ giới hạn trong khoảng [0, frame\_num-1].

+ Tiến hành copy các khung trang hiện tại cho trang tiếp theo, nếu trang tiếp theo chưa phải trang cuối cùng (j<n-1).

+ Tăng j lên 1 đơn vị (chuyển đến trang tiếp theo).

+ Thực hiện vòng lặp cho đến khi j duyệt qua hết n phần tử trong chuỗi.

* Xuất ouput gồm bảng total\_frames[][] và các lỗi trang.
* Kết thúc chương trình.
  1. **Trình bày tính đúng đắn của lưu đồ bằng cách chạy tay ít nhất 01 test case tối thiểu 10 phần tử**

1. ***Test case:***

INPUT

seq[10]={7,0,1,2,0,3,0,4,2,3}

n=10

frame\_num=3

OUTPUT (Kết quả giải tay)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **7** | **0** | **1** | **2** | **0** | **3** | **0** | **4** | **2** | **3** |
| 7 | 7 | 7 | 2 | 2 | 2 | 2 | 4 | 4 | 4 |
|  | 0 | 0 | 0 | 0 | 3 | 3 | 3 | 2 | 2 |
|  |  | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 3 |
| **\*** | **\*** | **\*** | **\*** |  | **\*** | **\*** | **\*** | **\*** | **\*** |

Number of Page Fault: 9

1. ***Chứng minh tính đúng đắn của lưu đồ bằng chạy tay test case trên:***

* Khai báo biến i, j, replace\_i, fault\_num
* Khởi tạo:
  + Mảng total\_frames[3][10] =[[None\*10]\*3]
  + Mảng faults[10]=[0\*10]
  + j=0, replace\_i=0, fault\_num=0
* Sử dụng vòng lặp cho đến khi j>=n (n=10):
  + Ở vòng lặp j=0 :
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][0] không chứa 7 (seq[0]).
    - total\_frames[0][0]=seq[0]=7
    - replace\_i=(0+1)%3=1
    - Đánh dấu trang j=0 có lỗi: faults[0]=1, tăng số lỗi fault\_num=1
    - Ta kiểm tra thấy j<n-1(0<9) nên copy khung trang cho trang 1 tiếp theo: total\_frames[i][1]=total\_frames[i][0], i [0, 2]
    - Lúc này, total\_frames[i][0]=total\_frames[i][1]=[7,None,None], i [0, 2]
    - j=1
  + Ở vòng lặp j=1:
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][1] không chứa 0 (seq[1]).
    - total\_frames[1][1]=seq[1]=0
    - replace\_i=(1+1)%3=2
    - Đánh dấu trang j=1 có lỗi: faults[1]=1, tăng số lỗi fault\_num=2
    - Ta kiểm tra thấy j<n-1(1<9) nên copy khung trang cho trang 2 tiếp theo: total\_frames[i][2]=total\_frames[i][1], i [0, 2]
    - Lúc này, total\_frames[i][1]= total\_frames[i][2]=[7,0,None], i [0, 2]
    - j=2
  + Ở vòng lặp j=2:
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][2] không chứa 1
    - total\_frames[2][2]=seq[2]=1
    - replace\_i=(2+1)%3=0
    - Đánh dấu trang j=2 có lỗi: faults[2]=1, tăng số lỗi fault\_num=3
    - Ta kiểm tra thấy j<n-1(2<9) nên copy khung trang cho trang 3 tiếp theo: total\_frames[i][3]=total\_frames[i][2], i [0, 2]
    - Lúc này, total\_frames[i][2]=total\_frames[i][3]=[7,0,1], i [0, 2]
    - j=3
  + Ở vòng lặp j=3:
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][3] không chứa 2
    - total\_frames[0][3]=seq[3]=2
    - replace\_i=(0+1)%3=1
    - Đánh dấu trang j=3 có lỗi: faults[3]=1, tăng số lỗi fault\_num=4
    - Ta kiểm tra thấy j<n-1(3<9) nên copy khung trang cho trang 4 tiếp theo: total\_frames[i][4]=total\_frames[i][3], i [0, 2]
    - Lúc này, total\_frames[i][3]=total\_frames[i][4]=[2,0,1], i [0, 2]
    - j=4
  + Ở vòng lặp j=4:
    - Với i [0, 2], ta kiểm tra thấy total\_frames[1][4]==0
    - Ta kiểm tra thấy j<n-1(4<9) nên copy khung trang cho trang 5 tiếp theo: total\_frames[i][5]=total\_frames[i][4], i [0, 2]
    - Lúc này, total\_frames[i][4]=total\_frames[i][5]=[2,0,1], i [0, 2]
    - j=5
  + Ở vòng lặp j=5:
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][5] không chứa 3
    - total\_frames[1][5]=seq[5]=3
    - replace\_i=(1+1)%3=2
    - Đánh dấu trang j=5 có lỗi: faults[5]=1, tăng số lỗi fault\_num=5
    - Ta kiểm tra thấy j<n-1(5<9) nên copy khung trang cho trang 6 tiếp theo: total\_frames[i][6]=total\_frames[i][5], i [0, 2]
    - Lúc này, total\_frames[i][5]=total\_frames[i][6]=[2,3,1], i [0, 2]
    - j=6
  + Ở vòng lặp j=6:
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][6] không chứa 0
    - total\_frames[2][6]=seq[6]=0
    - replace\_i=(2+1)%3=0
    - Đánh dấu trang j=6 có lỗi: faults[6]=1, tăng số lỗi fault\_num=6
    - Ta kiểm tra thấy j<n-1(6<9) nên copy khung trang cho trang 7 tiếp theo: total\_frames[i][7]=total\_frames[i][6], i [0, 2]
    - Lúc này, total\_frames[i][6]=total\_frames[i][7]=[2,3,0], i [0, 2]
    - j=7
  + Ở vòng lặp j=7:
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][7] không chứa 4
    - total\_frames[0][7]=seq[7]=4
    - replace\_i=(0+1)%3=1
    - Đánh dấu trang j=7 có lỗi: faults[7]=1, tăng số lỗi fault\_num=7
    - Ta kiểm tra thấy j<n-1(7<9) nên copy khung trang cho trang 8 tiếp theo: total\_frames[i][8]=total\_frames[i][7], i [0, 2]
    - Lúc này, total\_frames[i][7]=total\_frames[i][8]=[4,3,0], i [0, 2]
    - j=8
  + Ở vòng lặp j=8:
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][8] không chứa 2
    - total\_frames[1][8]=seq[8]=2
    - replace\_i=(1+1)%3=2
    - Đánh dấu trang j=8 có lỗi: faults[8]=1, tăng số lỗi fault\_num=8
    - Ta kiểm tra thấy j<n-1(8<9) nên copy khung trang cho trang 9 tiếp theo: total\_frames[i][9]=total\_frames[i][8], i [0, 2]
    - Lúc này, total\_frames[i][8]=total\_frames[i][9]=[4,2,0], i [0, 2]
    - j=9
  + Ở vòng lặp j=9
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][9] không chứa 3
    - total\_frames[2][9]=seq[9]=3
    - replace\_i=(2+1)%3=0
    - Đánh dấu trang j=9 có lỗi: faults[9]=1, tăng số lỗi fault\_num=9
    - Ta kiểm tra thấy j==n-1(9==9)
    - Lúc này, total\_frames[i][9]=[4,2,3], i [0, 2]
    - j=10
  + Ở vòng lặp j=10 kiểm tra thấy j==n==10, thoát khỏi vòng lặp.
* Xuất output.
* Kết thúc chương trình.

1. ***Minh họa output chạy tay trên lưu đồ:***

Number of Page Fault: 9 (fault\_num)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **7** | **0** | **1** | **2** | **0** | **3** | **0** | **4** | **2** | **3** |
| 7 | 7 | 7 | 2 | 2 | 2 | 2 | 4 | 4 | 4 |
| None | 0 | 0 | 0 | 0 | 3 | 3 | 3 | 2 | 2 |
| None | None | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 3 |
| **1** | **1** | **1** | **1** | **0** | **1** | **1** | **1** | **1** | **1** |

*Lưu ý*: Khi xuất ra màn hình, None được thay bằng khoảng trắng, và dòng cuối cùng của output (mảng faults[]) giá trị 1 được thay bằng “\*”, 0 được thay bằng khoảng trắng.

⮚ ***Kết quả chạy tay trên lưu đồ giống với kết quả giải tay, chứng tỏ lưu đồ đúng.***

* 1. **Thực hiện code cho giải thuật**
* Khai báo và khởi tạo giá trị cho các biến replace\_i, fault\_num, mảng faults[].
* In chuỗi tham chiếu ban đầu seq[] ra màn hình.
* Khởi tạo tất cả ô bảng total\_frames[][] với giá trị -1 (tương ứng với None trong phần 1.2).

Text

Description automatically generated

*Hình 6. Thực hiện code cho giải thuật FIFO (1)*

* Ở mỗi vòng lặp j (mỗi trang trong chuỗi):

+ Khởi tạo giá trị biến available=0 (mặc định không tìm thấy trong khung trang).

+ Lặp qua các khung trang hiện tại, nếu tìm thấy gán available=1.

+ Sau khi tìm hết trong các khung trang, nếu available vẫn = 0 (không tìm thấy), ta tiến hành thay trang.

* Gán trang hiện tại seq[j] cho ô total\_frames[replace\_i][j], khung trang tại vị trí replace\_i trong cột trang hiện tại.
* replace\_i tăng lên 1, vẫn trong khoảng [0, frame\_num-1]
* Đánh dấu trang j có lỗi (faults[j]=1), tăng số lỗi thêm 1.

+ Copy các khung trang hiện tại cho trang kế tiếp.

Text

Description automatically generated

*Hình 7. Thực hiện code cho giải thuật FIFO (2)*

* Xuất ra màn hình các giá trị trong bảng total\_frames[][], với giá trị bằng -1 (None/rỗng) xuất ra khoảng trắng thay thế.
* Xuất ra màn hình các đánh dấu lỗi trong mảng faults[], với giá trị 1 (có lỗi) xuất ra kí tự “\*”, với giá trị 0 xuất ra khoảng trắng.
* Xuất ra số lỗi trang lưu trong biến fault\_num.

Text

Description automatically generated

*Hình 8. Thực hiện code cho giải thuật FIFO (3)*

* 1. **Trình bày tính đúng đắn của code bằng cách chạy ít nhất 03 test case**

*Testcase mặc định:*

INPUT

n=11

seq[]={2, 1, 5, 2, 2, 6, 7, 9, 0, 0, 7}

frame\_num=3

OUTPUT (Kết quả giải tay)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **2** | **1** | **5** | **2** | **2** | **6** | **7** | **9** | **0** | **0** | **7** |
| 2 | 2 | 2 | 2 | 2 | 6 | 6 | 6 | 0 | 0 | 0 |
|  | 1 | 1 | 1 | 1 | 1 | 7 | 7 | 7 | 7 | 7 |
|  |  | 5 | 5 | 5 | 5 | 5 | 9 | 9 | 9 | 9 |
| **\*** | **\*** | **\*** |  |  | **\*** | **\*** | **\*** | **\*** |  |  |

Number of Page Fault: 7

Text

Description automatically generated

*Hình 9. Chạy test case mặc định bằng code giải thuật FIFO*

⮚ ***Kết quả thực hiện bằng code giống với kết quả giải tay.***

*Testcase nhập tay 1:*

INPUT

n=20

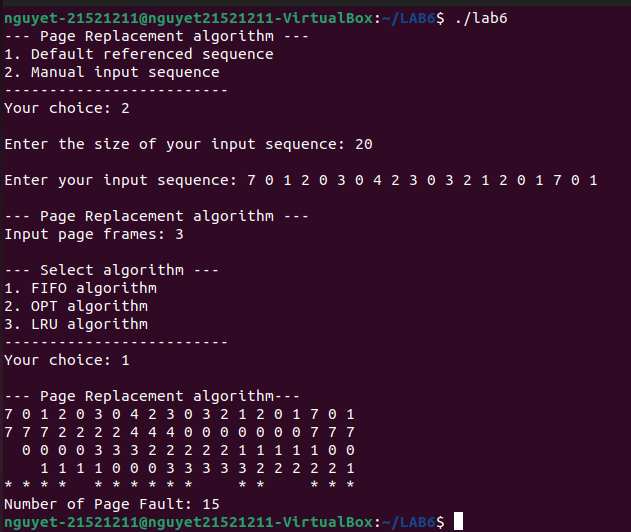
seq[]={7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1}

frame\_num=3

OUTPUT (Kết quả giải tay)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **7** | **0** | **1** | **2** | **0** | **3** | **0** | **4** | **2** | **3** | **0** | **3** | **2** | **1** | **2** | **0** | **1** | **7** | **0** | **1** |
| 7 | 7 | 7 | 2 | 2 | 2 | 2 | 4 | 4 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 7 | 7 | 7 |
|  | 0 | 0 | 0 | 0 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
|  |  | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 3 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 1 |
| **\*** | **\*** | **\*** | **\*** |  | **\*** | **\*** | **\*** | **\*** | **\*** | **\*** |  |  | **\*** | **\*** |  |  | **\*** | **\*** | **\*** |

Number of Page Fault: 15



*Hình 10. Chạy test case nhập tay 1 bằng code giải thuật FIFO*

⮚ ***Kết quả thực hiện bằng code giống với kết quả giải tay.***

*Testcase nhập tay 2:*

INPUT

n=20

seq[]={1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6}

frame\_num=3

OUTPUT (Kết quả giải tay)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **2** | **1** | **5** | **6** | **2** | **1** | **2** | **3** | **7** | **6** | **3** | **2** | **1** | **2** | **3** | **6** |
| 1 | 1 | 1 | 4 | 4 | 4 | 4 | 6 | 6 | 6 | 6 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 6 |
|  | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 7 | 7 | 7 | 7 | 1 | 1 | 1 | 1 |
|  |  | 3 | 3 | 3 | 3 | 5 | 5 | 5 | 1 | 1 | 1 | 1 | 6 | 6 | 6 | 6 | 6 | 3 | 3 |
| **\*** | **\*** | **\*** | **\*** |  | **\*** | **\*** | **\*** | **\*** | **\*** |  | **\*** | **\*** | **\*** |  | **\*** | **\*** |  | **\*** | **\*** |

Number of Page Fault: 16

Text

Description automatically generated

*Hình 11. Chạy test case nhập tay 2 bằng code giải thuật FIFO*

⮚ ***Kết quả thực hiện bằng code giống với kết quả giải tay.***

⮚ ***Cả 3 test case đều cho kết quả giống với kết quả giải tay. Kết luận code đúng.***

***Ex 2.Giải thuật OPT***

* 1. **Vẽ lưu đồ giải thuật**

Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

*Hình 12. Lưu đồ giải thuật OPT*

* Khởi tạo biến fault\_num=0 và tất cả giá trị trong mảng faults[]=0, do ban đầu chưa có lỗi trang nào.
* replace\_i khởi tạo=0: vị trí khung trang đầu tiên thêm vào.
* Khởi tạo mảng options[frame\_num] là mảng chứa các lựa chọn để thay trang, lưu vị trí ***tương lai*** gần nhất trong chuỗi tham chiếu của các trang đang chiếm chỗ trong khung trang. Từ đó, lựa chọn được trang có vị trí xa nhất (index lớn nhất). Giá trị ban đầu của tất cả các phần tử trong options[] là n (nếu trang không được tham chiếu trong tương lai, giá trị của nó trong options[] vẫn = n, lớn hơn tất cả các trang được tham chiếu lần nữa (luôn <= n-1)).
* *Ví dụ: Ta có chuỗi tham chiếu {…,6,7,4,..,5}*

*Khi trang thứ 4 (trang số 7) được tham chiếu, khung trang đầy, ta tiến hành lưu những vị trí tham chiếu tương lai của các trang trong khung trang (4, 5, 6) vào mảng options[]. Ban đầu, khởi tạo options[]=[n, n, n]. Ta giả sử trang số 6 không được tham chiếu nữa. Sau khi tìm trong chuỗi, options[]=[5, n-1, n] tương ứng với khung trang 0, 1, 2 và các trang số 4, 5, 6. Khi đó, tìm được giá trị lớn nhất trong options[] là n, vị trí khung trang chọn để thay tương ứng là vị trí 2 (thay trang số 6 ra).*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *index* | ***…*** | ***3*** | ***4*** | ***5*** | ***…*** | ***n-1*** |
|  | ***…*** | ***6*** | ***7*** | ***4*** | ***…*** | ***5*** |
| ***0*** |  | *4* | *4* |  |  |  |
| ***1*** |  | *5* | *5* |  |  |  |
| ***2*** |  | *6* | ***7*** |  |  |  |

* Ta thực hiện vòng lặp bắt đầu j=0 cho đến n-1 (j đi qua từng phần tử của chuỗi tham chiếu). Tại mỗi phần tử:

+ Tiến hành tìm trang tại vị trí j, xem có trong các khung trang hiện tại không. Nếu có, không thực hiện thay trang. Ngược lại thực hiện thay trang:

* Kiểm tra còn khung trang trống không. Nếu có, thêm trực tiếp vào vị trí còn trống nhỏ nhất (replace\_i).
* Nếu không còn khung trang trống, thực hiện tìm vị trí tham chiếu gần nhất trong tương lai của từng trang trong khung trang hiện tại, lưu vào mảng options[]. Trong mảng options[], tìm giá trị lớn nhất (vị trí tham chiếu xa nhất), chọn khung trang tương ứng với giá trị này trong options[] để thay (replace\_i=i\_max). Reset mảng options[]={n}, replace\_i=frame\_num để tránh nhầm trường hợp khung trang còn trống.
* Đánh dấu trang thứ j có lỗi, tăng lỗi trang thêm 1.

+ Tiến hành copy các khung trang hiện tại cho trang tiếp theo, nếu trang tiếp theo chưa phải trang cuối cùng (j<n-1).

+ Tăng j lên 1 đơn vị (chuyển đến trang tiếp theo).

+ Thực hiện vòng lặp cho đến khi j duyệt qua hết n phần tử trong chuỗi.

* Xuất ouput gồm bảng total\_frames[][] và các lỗi trang.
* Kết thúc chương trình.
  1. **Trình bày tính đúng đắn của lưu đồ bằng cách chạy tay ít nhất 01 test case tối thiểu 10 phần tử**
     + - 1. ***Test case:***

INPUT

seq[10]={7,0,1,2,0,3,0,4,2,3}

n=10

frame\_num=3

OUTPUT (Kết quả giải tay)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **7** | **0** | **1** | **2** | **0** | **3** | **0** | **4** | **2** | **3** |
| 7 | 7 | 7 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 4 | 4 |
|  |  | 1 | 1 | 1 | 3 | 3 | 3 | 3 | 3 |
| **\*** | **\*** | **\*** | **\*** |  | **\*** |  | **\*** |  |  |

Number of Page Fault: 6

* + - * 1. ***Chứng minh tính đúng đắn của lưu đồ bằng chạy tay test case trên:***
* Khai báo biến i, j, replace\_i, fault\_num
* Khởi tạo:
  + Mảng total\_frames[3][10] =[[None\*10]\*3]
  + Mảng faults[10]=[0\*10]
  + Mảng options[3]=[10\*3]
  + j=0, replace\_i=0, fault\_num=0
* Sử dụng vòng lặp cho đến khi j>=n (n=10):
  + Ở vòng lặp j=0 :
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][0] không chứa 7 (seq[0]).
    - Kiểm tra tiếp thấy khung trang còn trống (replace\_i=0<3), vì vậy total\_frames[0][0]=seq[0]=7, replace\_i=1
    - Đánh dấu trang j=0 có lỗi: faults[0]=1, tăng số lỗi fault\_num=1
    - Ta kiểm tra thấy j<n-1(0<9) nên copy khung trang cho trang 1 tiếp theo: total\_frames[i][1]=total\_frames[i][0], i [0, 2]
    - Lúc này, total\_frames[i][0]=total\_frames[i][1]=[7,None,None], i [0, 2]
    - j=1
  + Ở vòng lặp j=1:
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][1] không chứa 0 (seq[1]).
    - Kiểm tra tiếp thấy khung trang còn trống (replace\_i=1<3), vì vậy total\_frames[1][1]=seq[1]=0, replace\_i=2
    - Đánh dấu trang j=1 có lỗi: faults[1]=1, tăng số lỗi fault\_num=2
    - Ta kiểm tra thấy j<n-1(1<9) nên copy khung trang cho trang 2 tiếp theo: total\_frames[i][2]=total\_frames[i][1], i [0, 2]
    - Lúc này, total\_frames[i][1]= total\_frames[i][2]=[7,0,None], i [0, 2]
    - j=2
  + Ở vòng lặp j=2:
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][2] không chứa 1
    - Kiểm tra tiếp thấy khung trang còn trống (replace\_i=2<3), vì vậy total\_frames[2][2]=seq[2]=1, replace\_i=3
    - Đánh dấu trang j=2 có lỗi: faults[2]=1, tăng số lỗi fault\_num=3
    - Ta kiểm tra thấy j<n-1(2<9) nên copy khung trang cho trang 3 tiếp theo: total\_frames[i][3]=total\_frames[i][2], i [0, 2]
    - Lúc này, total\_frames[i][2]=total\_frames[i][3]=[7,0,1], i [0, 2]
    - j=3
  + Ở vòng lặp j=3:
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][3] không chứa 2
    - Kiểm tra tiếp thấy trong khung trang hết chỗ (replace\_i==3)
    - Tìm vị trí gần nhất trong tương lai của các trang đang có trong khung trang total\_frames[i][3], i [0, 2] và lưu vào mảng options[]. Mảng options[]=[10,4,10] do trang 0 (khung trang 1) được tham chiếu lần nữa tại vị trí thứ 4 trong chuỗi, trang 7 và 1 không được tham chiếu nữa.
    - Tìm vị trí của phần tử lớn nhất trong options[] (phần tử 10 ở vị trí 0), lưu vị trí đó vào replace\_i(replace\_i=0): total\_frames[0][3]=seq[3]=2
    - Reset options[]=[10, 10, 10], replace\_i=frame\_num=3
    - Đánh dấu trang j=3 có lỗi: faults[3]=1, tăng số lỗi fault\_num=4
    - Ta kiểm tra thấy j<n-1(3<9) nên copy khung trang cho trang 4 tiếp theo: total\_frames[i][4]=total\_frames[i][3], i [0, 2]
    - Lúc này, total\_frames[i][3]=total\_frames[i][4]=[2,0,1], i [0, 2]
    - j=4
  + Ở vòng lặp j=4:
    - Với i [0, 2], ta kiểm tra thấy total\_frames[1][4]==0
    - Ta kiểm tra thấy j<n-1(4<9) nên copy khung trang cho trang 5 tiếp theo: total\_frames[i][5]=total\_frames[i][4], i [0, 2]
    - Lúc này, total\_frames[i][4]=total\_frames[i][5]=[2,0,1], i [0, 2]
    - j=5
  + Ở vòng lặp j=5:
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][5] không chứa 3
    - Kiểm tra tiếp thấy trong khung trang hết chỗ (replace\_i==3)
    - Tìm vị trí gần nhất trong tương lai của các trang đang có trong khung trang total\_frames[i][5], i [0, 2] và lưu vào mảng options[]. Mảng options[]=[8,6,10].
    - Tìm vị trí của phần tử lớn nhất trong options[] (phần tử 10 ở vị trí 2), lưu vị trí đó vào replace\_i(replace\_i=2): total\_frames[2][5]=seq[5]=3
    - Reset options[]=[10, 10, 10], replace\_i=frame\_num=3
    - Đánh dấu trang j=5 có lỗi: faults[5]=1, tăng số lỗi fault\_num=5
    - Ta kiểm tra thấy j<n-1(5<9) nên copy khung trang cho trang 6 tiếp theo: total\_frames[i][6]=total\_frames[i][5], i [0, 2]
    - Lúc này, total\_frames[i][5]=total\_frames[i][6]=[2,0,3], i [0, 2]
    - j=6
  + Ở vòng lặp j=6:
    - Với i [0, 2], ta kiểm tra thấy total\_frames[1][6]==0
    - Ta kiểm tra thấy j<n-1(6<9) nên copy khung trang cho trang 7 tiếp theo: total\_frames[i][7]=total\_frames[i][6], i [0, 2]
    - Lúc này, total\_frames[i][6]=total\_frames[i][7]=[2,0,3], i [0, 2]
    - j=7
  + Ở vòng lặp j=7:
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][7] không chứa 4
    - Kiểm tra tiếp thấy trong khung trang hết chỗ (replace\_i==3)
    - Tìm vị trí gần nhất trong tương lai của các trang đang có trong khung trang total\_frames[i][5], i [0, 2] và lưu vào mảng options[]. Mảng options[]=[8,10,9].
    - Tìm vị trí của phần tử lớn nhất trong options[] (phần tử 10 ở vị trí 1), lưu vị trí đó vào replace\_i(replace\_i=1): total\_frames[1][7]=seq[7]=4
    - Reset options[]=[10, 10, 10], replace\_i=frame\_num=3
* Đánh dấu trang j=7 có lỗi: faults[7]=1, tăng số lỗi fault\_num=6
  + - Ta kiểm tra thấy j<n-1(7<9) nên copy khung trang cho trang 8 tiếp theo: total\_frames[i][8]=total\_frames[i][7], i [0, 2]
    - Lúc này, total\_frames[i][7]=total\_frames[i][8]=[2,4,3], i [0, 2]
    - j=8
  + Ở vòng lặp j=8:
    - Với i [0, 2], ta kiểm tra thấy total\_frames[0][8]==2
    - Ta kiểm tra thấy j<n-1(8<9) nên copy khung trang cho trang 9 tiếp theo: total\_frames[i][9]=total\_frames[i][8], i [0, 2]
    - Lúc này, total\_frames[i][8]=total\_frames[i][9]=[2,4,3], i [0, 2]
    - j=9
  + Ở vòng lặp j=9
    - Với i [0, 2], ta kiểm tra thấy total\_frames[2][9]==3
    - Ta kiểm tra thấy j==n-1(9==9)
    - Lúc này, total\_frames[i][9]=[2,4,3], i [0, 2]
    - j=10
  + Ở vòng lặp j=10 kiểm tra thấy j==n==10, thoát khỏi vòng lặp.
* Xuất output.
* Kết thúc chương trình.

1. ***Minh họa output chạy tay trên lưu đồ:***

Number of Page Fault: 6 (fault\_num)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **7** | **0** | **1** | **2** | **0** | **3** | **0** | **4** | **2** | **3** |
| 7 | 7 | 7 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| None | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 4 | 4 |
| None | None | 1 | 1 | 1 | 3 | 3 | 3 | 3 | 3 |
| **1** | **1** | **1** | **1** | **0** | **1** | **0** | **1** | **0** | **0** |

*Lưu ý*: Khi xuất ra màn hình, None được thay bằng khoảng trắng, và dòng cuối cùng của output (mảng faults[]) giá trị 1 được thay bằng “\*”, 0 được thay bằng khoảng trắng.

⮚ ***Kết quả chạy tay trên lưu đồ giống với kết quả giải tay, chứng tỏ lưu đồ đúng.***

* 1. **Thực hiện code cho giải thuật**

Text

Description automatically generated

*Hình 13. Thực hiện code cho giải thuật OPT (1)*

* Khai báo và khởi tạo giá trị cho các biến replace\_i, fault\_num, mảng faults[].
* In chuỗi tham chiếu ban đầu seq[] ra màn hình.
* Khởi tạo tất cả ô bảng total\_frames[][] với giá trị -1 (tương ứng với None trong phần 2.2). options[]={n}

Text

Description automatically generated

*Hình 14. Thực hiện code cho giải thuật OPT (2)*

* Ở mỗi vòng lặp j (mỗi trang trong chuỗi):

+ Khởi tạo giá trị biến available=0 (mặc định không tìm thấy trong khung trang).

+ Lặp qua các khung trang hiện tại, nếu tìm thấy gán available=1.

+ Sau khi tìm hết trong các khung trang, nếu available vẫn = 0 (không tìm thấy), ta tiến hành thay trang.

* Kiểm tra khung trang còn trống (replace\_i<frame\_num). Nếu còn trống thêm trực tiếp vào vị trí replace\_i, replace\_i tăng lên 1. Nếu không còn trống:
* Đối với mỗi trang trong khung trang i [0, frame\_num-1], lặp qua tất cả các trang tương lai trong trong chuỗi (từ vị trí j+1 trở đi), tìm thấy tại vị trí nào thì lưu vào options[i]. Chỉ tìm vị trí gần nhất, dùng break dừng ngay khi tìm thấy vị trí đầu tiên.

Text

Description automatically generated

*Hình 15. Thực hiện code cho giải thuật OPT (3)*

* Trong mảng options[] tìm phần tử có giá trị lớn nhất, lưu index của nó vào replace\_i.
* Thay trang tại vị trí replace\_i: total\_frames[replace\_i][j]=seq[j]
* Reset mảng options[]={n} và replace\_i=frame\_num.
* Đánh dấu trang j có lỗi (faults[j]=1), tăng số lỗi thêm 1.

+ Copy các khung trang hiện tại cho trang kế tiếp.

* Xuất ra màn hình các giá trị trong bảng total\_frames[][], với giá trị bằng -1 (None/rỗng) xuất ra khoảng trắng thay thế.
* Xuất ra màn hình các đánh dấu lỗi trong mảng faults[], với giá trị 1 (có lỗi) xuất ra kí tự “\*”, với giá trị 0 xuất ra khoảng trắng.
* Xuất ra số lỗi trang lưu trong biến fault\_num.

Text

Description automatically generated

*Hình 16. Thực hiện code cho giải thuật OPT (4)*

* 1. **Trình bày tính đúng đắn của code bằng cách chạy ít nhất 03 test case**

*Testcase mặc định:*

INPUT

n=11

seq[]={2, 1, 5, 2, 2, 6, 7, 9, 0, 0, 7}

frame\_num=3

OUTPUT (Kết quả giải tay)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **2** | **1** | **5** | **2** | **2** | **6** | **7** | **9** | **0** | **0** | **7** |
| 2 | 2 | 2 | 2 | 2 | 6 | 7 | 7 | 7 | 7 | 7 |
|  | 1 | 1 | 1 | 1 | 1 | 1 | 9 | 0 | 0 | 0 |
|  |  | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| **\*** | **\*** | **\*** |  |  | **\*** | **\*** | **\*** | **\*** |  |  |

Number of Page Fault: 7

Text

Description automatically generated

*Hình 17. Chạy test case mặc định bằng code giải thuật OPT*

⮚ ***Kết quả thực hiện bằng code giống với kết quả giải tay.***

*Testcase nhập tay 1:*

INPUT

n=20

seq[]={7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1}

frame\_num=3

OUTPUT (Kết quả giải tay)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **7** | **0** | **1** | **2** | **0** | **3** | **0** | **4** | **2** | **3** | **0** | **3** | **2** | **1** | **2** | **0** | **1** | **7** | **0** | **1** |
| 7 | 7 | 7 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 7 | 7 | 7 |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 4 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  |  | 1 | 1 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **\*** | **\*** | **\*** | **\*** |  | **\*** |  | **\*** |  |  | **\*** |  |  | **\*** |  |  |  | **\*** |  |  |

Number of Page Fault: 9

Text

Description automatically generated

*Hình 18. Chạy test case nhập tay 1 bằng code giải thuật OPT*

⮚ ***Kết quả thực hiện bằng code giống với kết quả giải tay.***

*Testcase nhập tay 2:*

INPUT

n=20

seq[]={1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6}

frame\_num=3

OUTPUT (Kết quả giải tay)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **2** | **1** | **5** | **6** | **2** | **1** | **2** | **3** | **7** | **6** | **3** | **2** | **1** | **2** | **3** | **6** |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 6 |
|  | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 7 | 7 | 7 | 2 | 2 | 2 | 2 | 2 |
|  |  | 3 | 4 | 4 | 4 | 5 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 1 | 1 | 1 | 1 |
| **\*** | **\*** | **\*** | **\*** |  |  | **\*** | **\*** |  |  |  | **\*** | **\*** |  |  | **\*** | **\*** |  |  | **\*** |

Number of Page Fault: 11

Text

Description automatically generated

*Hình 19. Chạy test case nhập tay 2 bằng code giải thuật OPT*

⮚ ***Kết quả thực hiện bằng code giống với kết quả giải tay.***

⮚ ***Cả 3 test case đều cho kết quả giống với kết quả giải tay. Kết luận code đúng.***

***Ex 3.Giải thuật LRU***

* 1. **Vẽ lưu đồ giải thuật**

Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

*Hình 20. Lưu đồ giải thuật LRU*

* Khởi tạo biến fault\_num=0 và tất cả giá trị trong mảng faults[]=0, do ban đầu chưa có lỗi trang nào.
* replace\_i khởi tạo=0: vị trí khung trang đầu tiên thêm vào.
* Khởi tạo mảng options[frame\_num] là mảng chứa các lựa chọn để thay trang, lưu vị trí ***quá khứ*** gần nhất trong chuỗi tham chiếu của các trang đang chiếm chỗ trong khung trang. Từ đó, lựa chọn được trang có thời điểm tham chiếu nhỏ nhất (index trong chuỗi tham chiếu nhỏ nhất). Giá trị ban đầu của tất cả các phần tử trong options[] là -1.
* Ta thực hiện vòng lặp bắt đầu j=0 cho đến n-1 (j đi qua từng phần tử của chuỗi tham chiếu). Tại mỗi phần tử:

+ Tiến hành tìm trang tại vị trí j, xem có trong các khung trang hiện tại không. Nếu có, không thực hiện thay trang. Ngược lại thực hiện thay trang:

* Kiểm tra còn khung trang trống không. Nếu có, thêm trực tiếp vào vị trí còn trống nhỏ nhất (replace\_i).
* Nếu không còn khung trang trống, thực hiện tìm vị trí tham chiếu gần nhất trong quá khứ của từng trang trong khung trang hiện tại, lưu vào mảng options[]. Trong mảng options[], tìm giá trị nhỏ nhất (thời điểm tham chiếu nhỏ nhất), chọn khung trang tương ứng với giá trị này trong options[] để thay (replace\_i=i\_min). Reset mảng options[]={-1}, replace\_i=frame\_num để tránh nhầm trường hợp khung trang còn trống.
* Đánh dấu trang thứ j có lỗi, tăng lỗi trang thêm 1.

+ Tiến hành copy các khung trang hiện tại cho trang tiếp theo, nếu trang tiếp theo chưa phải trang cuối cùng (j<n-1).

+ Tăng j lên 1 đơn vị (chuyển đến trang tiếp theo).

+ Thực hiện vòng lặp cho đến khi j duyệt qua hết n phần tử trong chuỗi.

* Xuất ouput gồm bảng total\_frames[][] và các lỗi trang.
* Kết thúc chương trình.
  1. **Trình bày tính đúng đắn của lưu đồ bằng cách chạy tay ít nhất 01 test case tối thiểu 10 phần tử**
     + - 1. ***Test case:***

INPUT

seq[10]={7,0,1,2,0,3,0,4,2,3}

n=10

frame\_num=3

OUTPUT (Kết quả giải tay)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **7** | **0** | **1** | **2** | **0** | **3** | **0** | **4** | **2** | **3** |
| 7 | 7 | 7 | 2 | 2 | 2 | 2 | 4 | 4 | 4 |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 |
|  |  | 1 | 1 | 1 | 3 | 3 | 3 | 2 | 2 |
| **\*** | **\*** | **\*** | **\*** |  | **\*** |  | **\*** | **\*** | **\*** |

Number of Page Fault: 8

* + - * 1. ***Chứng minh tính đúng đắn của lưu đồ bằng chạy tay test case trên:***
* Khai báo biến i, j, replace\_i, fault\_num
* Khởi tạo:
  + Mảng total\_frames[3][10] =[[None\*10]\*3]
  + Mảng faults[10]=[0\*10]
  + Mảng options[3]=[(-1)\*3]
  + j=0, replace\_i=0, fault\_num=0
* Sử dụng vòng lặp cho đến khi j>=n (n=10):
  + Ở vòng lặp j=0 :
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][0] không chứa 7 (seq[0]).
    - Kiểm tra tiếp thấy khung trang còn trống (replace\_i=0<3), vì vậy total\_frames[0][0]=seq[0]=7, replace\_i=1
    - Đánh dấu trang j=0 có lỗi: faults[0]=1, tăng số lỗi fault\_num=1
    - Ta kiểm tra thấy j<n-1(0<9) nên copy khung trang cho trang 1 tiếp theo: total\_frames[i][1]=total\_frames[i][0], i [0, 2]
    - Lúc này, total\_frames[i][0]=total\_frames[i][1]=[7,None,None], i [0, 2]
    - j=1
  + Ở vòng lặp j=1:
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][1] không chứa 0 (seq[1]).
    - Kiểm tra tiếp thấy khung trang còn trống (replace\_i=1<3), vì vậy total\_frames[1][1]=seq[1]=0, replace\_i=2
    - Đánh dấu trang j=1 có lỗi: faults[1]=1, tăng số lỗi fault\_num=2
    - Ta kiểm tra thấy j<n-1(1<9) nên copy khung trang cho trang 2 tiếp theo: total\_frames[i][2]=total\_frames[i][1], i [0, 2]
    - Lúc này, total\_frames[i][1]= total\_frames[i][2]=[7,0,None], i [0, 2]
    - j=2
  + Ở vòng lặp j=2:
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][2] không chứa 1
    - Kiểm tra tiếp thấy khung trang còn trống (replace\_i=2<3), vì vậy total\_frames[2][2]=seq[2]=1, replace\_i=3
    - Đánh dấu trang j=2 có lỗi: faults[2]=1, tăng số lỗi fault\_num=3
    - Ta kiểm tra thấy j<n-1(2<9) nên copy khung trang cho trang 3 tiếp theo: total\_frames[i][3]=total\_frames[i][2], i [0, 2]
    - Lúc này, total\_frames[i][2]=total\_frames[i][3]=[7,0,1], i [0, 2]
    - j=3
  + Ở vòng lặp j=3:
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][3] không chứa 2
    - Kiểm tra tiếp thấy trong khung trang hết chỗ (replace\_i==3)
    - Tìm vị trí gần nhất trong quá khứ của các trang đang có trong khung trang total\_frames[i][3], i [0, 2] và lưu vào mảng options[]. Mảng options[]=[0,1,2] do các trang 7,0,1 được tham chiếu gần nhất trong quá khứ tại vị trí 0,1,2 trong chuỗi.
    - Tìm vị trí của phần tử nhỏ nhất trong options[] (phần tử 0 ở vị trí 0), lưu vị trí đó vào replace\_i(replace\_i=0): total\_frames[0][3]=seq[3]=2
    - Reset options[]=[-1,-1,-1], replace\_i=frame\_num=3
    - Đánh dấu trang j=3 có lỗi: faults[3]=1, tăng số lỗi fault\_num=4
    - Ta kiểm tra thấy j<n-1(3<9) nên copy khung trang cho trang 4 tiếp theo: total\_frames[i][4]=total\_frames[i][3], i [0, 2]
    - Lúc này, total\_frames[i][3]=total\_frames[i][4]=[2,0,1], i [0, 2]
    - j=4
  + Ở vòng lặp j=4:
    - Với i [0, 2], ta kiểm tra thấy total\_frames[1][4]==0
    - Ta kiểm tra thấy j<n-1(4<9) nên copy khung trang cho trang 5 tiếp theo: total\_frames[i][5]=total\_frames[i][4], i [0, 2]
    - Lúc này, total\_frames[i][4]=total\_frames[i][5]=[2,0,1], i [0, 2]
    - j=5
  + Ở vòng lặp j=5:
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][5] không chứa 3
    - Kiểm tra tiếp thấy trong khung trang hết chỗ (replace\_i==3)
    - Tìm vị trí gần nhất trong quá khứ của các trang đang có trong khung trang total\_frames[i][5], i [0, 2] và lưu vào mảng options[]. Mảng options[]=[3,4,2].
    - Tìm vị trí của phần tử nhỏ nhất trong options[] (phần tử 2 ở vị trí 2), lưu vị trí đó vào replace\_i(replace\_i=2): total\_frames[2][5]=seq[5]=3
    - Reset options[]=[-1, -1, -1], replace\_i=frame\_num=3
    - Đánh dấu trang j=5 có lỗi: faults[5]=1, tăng số lỗi fault\_num=5
    - Ta kiểm tra thấy j<n-1(5<9) nên copy khung trang cho trang 6 tiếp theo: total\_frames[i][6]=total\_frames[i][5], i [0, 2]
    - Lúc này, total\_frames[i][5]=total\_frames[i][6]=[2,0,3], i [0, 2]
    - j=6
  + Ở vòng lặp j=6:
    - Với i [0, 2], ta kiểm tra thấy total\_frames[1][6]==0
    - Ta kiểm tra thấy j<n-1(6<9) nên copy khung trang cho trang 7 tiếp theo: total\_frames[i][7]=total\_frames[i][6], i [0, 2]
    - Lúc này, total\_frames[i][6]=total\_frames[i][7]=[2,0,3], i [0, 2]
    - j=7
  + Ở vòng lặp j=7:
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][7] không chứa 4
    - Kiểm tra tiếp thấy trong khung trang hết chỗ (replace\_i==3)
    - Tìm vị trí gần nhất trong quá khứ của các trang đang có trong khung trang total\_frames[i][7], i [0, 2] và lưu vào mảng options[]. Mảng options[]=[3,6,5].
    - Tìm vị trí của phần tử nhỏ nhất trong options[] (phần tử 3 ở vị trí 0), lưu vị trí đó vào replace\_i(replace\_i=0): total\_frames[0][7]=seq[7]=4
    - Reset options[]=[-1, -1, -1], replace\_i=frame\_num=3
* Đánh dấu trang j=7 có lỗi: faults[7]=1, tăng số lỗi fault\_num=6
  + - Ta kiểm tra thấy j<n-1(7<9) nên copy khung trang cho trang 8 tiếp theo: total\_frames[i][8]=total\_frames[i][7], i [0, 2]
    - Lúc này, total\_frames[i][7]=total\_frames[i][8]=[4,0,3], i [0, 2]
    - j=8
  + Ở vòng lặp j=8:
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][8] không chứa 2
    - Kiểm tra tiếp thấy trong khung trang hết chỗ (replace\_i==3)
    - Tìm vị trí gần nhất trong quá khứ của các trang đang có trong khung trang total\_frames[i][8], i [0, 2] và lưu vào mảng options[]. Mảng options[]=[7,6,5].
    - Tìm vị trí của phần tử nhỏ nhất trong options[] (phần tử 5 ở vị trí 2), lưu vị trí đó vào replace\_i(replace\_i=2): total\_frames[2][8]=seq[8]=2
    - Reset options[]=[-1, -1, -1], replace\_i=frame\_num=3
* Đánh dấu trang j=8 có lỗi: faults[8]=1, tăng số lỗi fault\_num=7
  + - Ta kiểm tra thấy j<n-1(8<9) nên copy khung trang cho trang 9 tiếp theo: total\_frames[i][9]=total\_frames[i][8], i [0, 2]
    - Lúc này, total\_frames[i][8]=total\_frames[i][9]=[4,0,2], i [0, 2]
    - j=9
  + Ở vòng lặp j=9
    - Với i [0, 2], ta kiểm tra thấy trong total\_frames[i][9] không chứa 3
    - Kiểm tra tiếp thấy trong khung trang hết chỗ (replace\_i==3)
    - Tìm vị trí gần nhất trong quá khứ của các trang đang có trong khung trang total\_frames[i][9], i [0, 2] và lưu vào mảng options[]. Mảng options[]=[7,6,8].
    - Tìm vị trí của phần tử nhỏ nhất trong options[][] (phần tử 8 ở vị trí 1), lưu vị trí đó vào replace\_i(replace\_i=1): total\_frames[1][9]=seq[9]=3
    - Reset options[]=[-1, -1, -1], replace\_i=frame\_num=3
* Đánh dấu trang j=9 có lỗi: faults[9]=1, tăng số lỗi fault\_num=8
  + - Ta kiểm tra thấy j==n-1(9==9)
    - Lúc này, total\_frames[i][9]=[4,3,2], i [0, 2]
    - j=10
  + Ở vòng lặp j=10 kiểm tra thấy j==n==10, thoát khỏi vòng lặp.
* Xuất output.
* Kết thúc chương trình.
  + - * 1. ***Minh họa output chạy tay trên lưu đồ:***

Number of Page Fault: 8 (fault\_num)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **7** | **0** | **1** | **2** | **0** | **3** | **0** | **4** | **2** | **3** |
| 7 | 7 | 7 | 2 | 2 | 2 | 2 | 4 | 4 | 4 |
| None | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 |
| None | None | 1 | 1 | 1 | 3 | 3 | 3 | 2 | 2 |
| **1** | **1** | **1** | **1** | **0** | **1** | **0** | **1** | **1** | **1** |

*Lưu ý*: Khi xuất ra màn hình, None được thay bằng khoảng trắng, và dòng cuối cùng của output (mảng faults[]) giá trị 1 được thay bằng “\*”, 0 được thay bằng khoảng trắng.

⮚ ***Kết quả chạy tay trên lưu đồ giống với kết quả giải tay, chứng tỏ lưu đồ đúng.***

* 1. **Thực hiện code cho giải thuật**

Text

Description automatically generated

*Hình 21. Thực hiện code cho giải thuật LRU (1)*

* Khai báo và khởi tạo giá trị cho các biến replace\_i, fault\_num, mảng faults[].
* In chuỗi tham chiếu ban đầu seq[] ra màn hình.
* Khởi tạo tất cả ô bảng total\_frames[][] với giá trị -1 (tương ứng với None trong phần 3.2). options[]={-1}

Text

Description automatically generated

*Hình 22. Thực hiện code cho giải thuật LRU (2)*

* Ở mỗi vòng lặp j (mỗi trang trong chuỗi):

+ Khởi tạo giá trị biến available=0 (mặc định không tìm thấy trong khung trang).

+ Lặp qua các khung trang hiện tại, nếu tìm thấy gán available=1.

+ Sau khi tìm hết trong các khung trang, nếu available vẫn = 0 (không tìm thấy), ta tiến hành thay trang.

* Kiểm tra khung trang còn trống (replace\_i<frame\_num). Nếu còn trống thêm trực tiếp vào vị trí replace\_i, replace\_i tăng lên 1. Nếu không còn trống:
* Đối với mỗi trang trong khung trang i [0, frame\_num-1], lặp qua tất cả các trang quá khứ trong trong chuỗi (từ vị trí j-1 ngược về trước), tìm thấy tại vị trí nào thì lưu vào options[i]. Chỉ tìm vị trí gần nhất, dùng break dừng ngay khi tìm thấy vị trí đầu tiên.
* Trong mảng options[] tìm phần tử có giá trị nhỏ nhất, lưu index của nó vào replace\_i.
* Thay trang tại vị trí replace\_i: total\_frames[replace\_i][j]=seq[j]
* Reset mảng options[]={-1} và replace\_i=frame\_num.

Text

Description automatically generated

*Hình 23. Thực hiện code cho giải thuật LRU (3)*

* Đánh dấu trang j có lỗi (faults[j]=1), tăng số lỗi thêm 1.

+ Copy các khung trang hiện tại cho trang kế tiếp.

* Xuất ra màn hình các giá trị trong bảng total\_frames[][], với giá trị bằng -1 (None/rỗng) xuất ra khoảng trắng thay thế.
* Xuất ra màn hình các đánh dấu lỗi trong mảng faults[], với giá trị 1 (có lỗi) xuất ra kí tự “\*”, với giá trị 0 xuất ra khoảng trắng.
* Xuất ra số lỗi trang lưu trong biến fault\_num.
  1. **Trình bày tính đúng đắn của code bằng cách chạy ít nhất 03 test case**

*Testcase mặc định:*

INPUT

n=11

seq[]={2, 1, 5, 2, 2, 6, 7, 9, 0, 0, 7}

frame\_num=3

OUTPUT (Kết quả giải tay)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **2** | **1** | **5** | **2** | **2** | **6** | **7** | **9** | **0** | **0** | **7** |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 9 | 9 | 9 | 9 |
|  | 1 | 1 | 1 | 1 | 6 | 6 | 6 | 0 | 0 | 0 |
|  |  | 5 | 5 | 5 | 5 | 7 | 7 | 7 | 7 | 7 |
| **\*** | **\*** | **\*** |  |  | **\*** | **\*** | **\*** | **\*** |  |  |

Number of Page Fault: 7

Text

Description automatically generated

*Hình 24. Chạy test case mặc định bằng code giải thuật LRU*

⮚ ***Kết quả thực hiện bằng code giống với kết quả giải tay.***

*Testcase nhập tay 1:*

INPUT

n=20

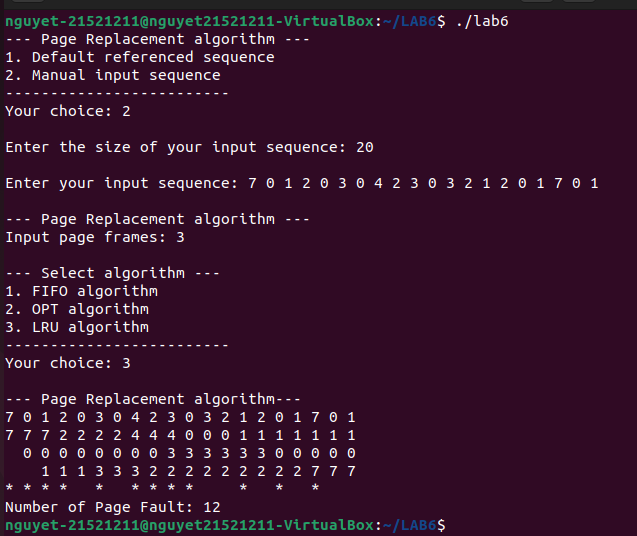
seq[]={7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1}

frame\_num=3

OUTPUT (Kết quả giải tay)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **7** | **0** | **1** | **2** | **0** | **3** | **0** | **4** | **2** | **3** | **0** | **3** | **2** | **1** | **2** | **0** | **1** | **7** | **0** | **1** |
| 7 | 7 | 7 | 2 | 2 | 2 | 2 | 4 | 4 | 4 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 3 | 3 | 3 | 3 | 3 | 0 | 0 | 0 | 0 | 0 |
|  |  | 1 | 1 | 1 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 7 | 7 | 7 |
| **\*** | **\*** | **\*** | **\*** |  | **\*** |  | **\*** | **\*** | **\*** | **\*** |  |  | **\*** |  | **\*** |  | **\*** |  |  |

Number of Page Fault: 12



*Hình 25. Chạy test case nhập tay 1 bằng code giải thuật LRU*

⮚ ***Kết quả thực hiện bằng code giống với kết quả giải tay.***

*Testcase nhập tay 2:*

INPUT

n=20

seq[]={1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6}

frame\_num=3

OUTPUT (Kết quả giải tay)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **2** | **1** | **5** | **6** | **2** | **1** | **2** | **3** | **7** | **6** | **3** | **2** | **1** | **2** | **3** | **6** |
| 1 | 1 | 1 | 4 | 4 | 4 | 5 | 5 | 5 | 1 | 1 | 1 | 7 | 7 | 7 | 2 | 2 | 2 | 2 | 2 |
|  | 2 | 2 | 2 | 2 | 2 | 2 | 6 | 6 | 6 | 6 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
|  |  | 3 | 3 | 3 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 6 | 6 | 6 | 1 | 1 | 1 | 6 |
| **\*** | **\*** | **\*** | **\*** |  | **\*** | **\*** | **\*** | **\*** | **\*** |  | **\*** | **\*** | **\*** |  | **\*** | **\*** |  |  | **\*** |

Number of Page Fault: 15

Text

Description automatically generated

*Hình 26. Chạy test case nhập tay 2 bằng code giải thuật LRU*

⮚ ***Kết quả thực hiện bằng code giống với kết quả giải tay.***

⮚ ***Cả 3 test case đều cho kết quả giống với kết quả giải tay. Kết luận code đúng.***

**Section 6.5**

***Ex 1. Nghịch lý Belady***

* 1. **Khái niệm nghịch lý Belady**

Nghịch lý Belady là số page fault (lỗi trang) tăng mặc dù quá trình đã được cấp nhiều frame (khung trang) hơn lúc ban đầu.

* 1. **Sử dụng chương trình đã viết chứng minh nghịch lý Belady**

Test case:

n=12

seq[]={1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5}

*Trường hợp 1: frame\_num=3*

Text

Description automatically generated

*Hình 27. Chạy test case với frame\_num=3*

Có thể thấy, khi sử dụng 3 khung trang, sẽ có 9 lỗi trang phát sinh.

*Trường hợp 2: frame\_num=4*

Text

Description automatically generated

*Hình 28. Chạy test case với frame\_num=4*

Với cùng một chuỗi tham chiếu, khi sử dụng 4 khung trang, ta thấy có 10 lỗi trang phát sinh.

Như vậy, số lỗi trang đã tăng lên mặc dù quá trình được cấp nhiều khung trang hơn lúc ban đầu.

⇨ Nghịch lý Belady.

***Ex 2. Mức độ hiệu quả và tính khả thi của 3 giải thuật***

**2.1. Giải thuật bất khả thi nhất**

Giải thuật bất khả thi nhất là giải thuật OPT. Vì rất khó để biết được trang nào sẽ được sử dụng trong tương lai và trang nào không được sử dụng.

**2.2. Giải thuật phức tạp nhất**

Giải thuật phức tạp nhất là giải thuật LRU. Vì phải ghi nhận lại các thời điểm các trang đó truy xuất và việc này vô cùng phức tạp đòi hỏi phải có sự hỗ trợ của phần cứng để ghi nhận lại thời điểm truy xuất và cập nhật của các trang đó khi nào.