Don Bosco Institute of Technology,Kurla
Academic Year 2019-20

EXPERIMENT NO: 6

Title: A program to simulate cache memory management using page replacement algorithms

Class:S.E Comps(Sem IV) Lecturer:Sejal.Chopra
Subject: PA Lab

EXPERIMENT NO:6

Page replacement algorithms

AIM	Writeaprogram to simulate cache memory management usingpagereplacementalgorithms.
LEARNING OBJECTIVE	To implement various page replacement policies.
LEARNING OUTCOME	Student will be able to visualize the scenario when new pages enter the cache memory using various algorithm.
LAB OUTCOME	CSL 403.1: Ability to compile a code for computer operations.
PROGRAM OUTCOME	PO11, PO52, PO83, PO9-3,PO12-2, PSO1-2
BLOOM'S TAXONOMY LEVEL	Remember, Understand
THEORY	In operating systems, whenever a new page is referred and not present in memory, page fault occurs and Operating System replaces one of the existing pages with newly needed page. Different page replacement algorithms suggest different ways to decide which page to replace. The target for all algorithms is to reduce number of page faults. In this algorithm, OS replaces the page that will not be used for the longest period of time in future. Advantages of Optimal Page Replacement Algorithm are as follows: 1) It is less complex and easy to implement. 2) A page is replaced with minimum fuss. 3) Simple data structures are used for this purpose. Disadvantages of Optimal Replacement Algorithm are as follows: 1) Not all operating systems can implement this algorithm. 2) Error detection is harder. 3) Least recently used page will be replaced which may sometimes

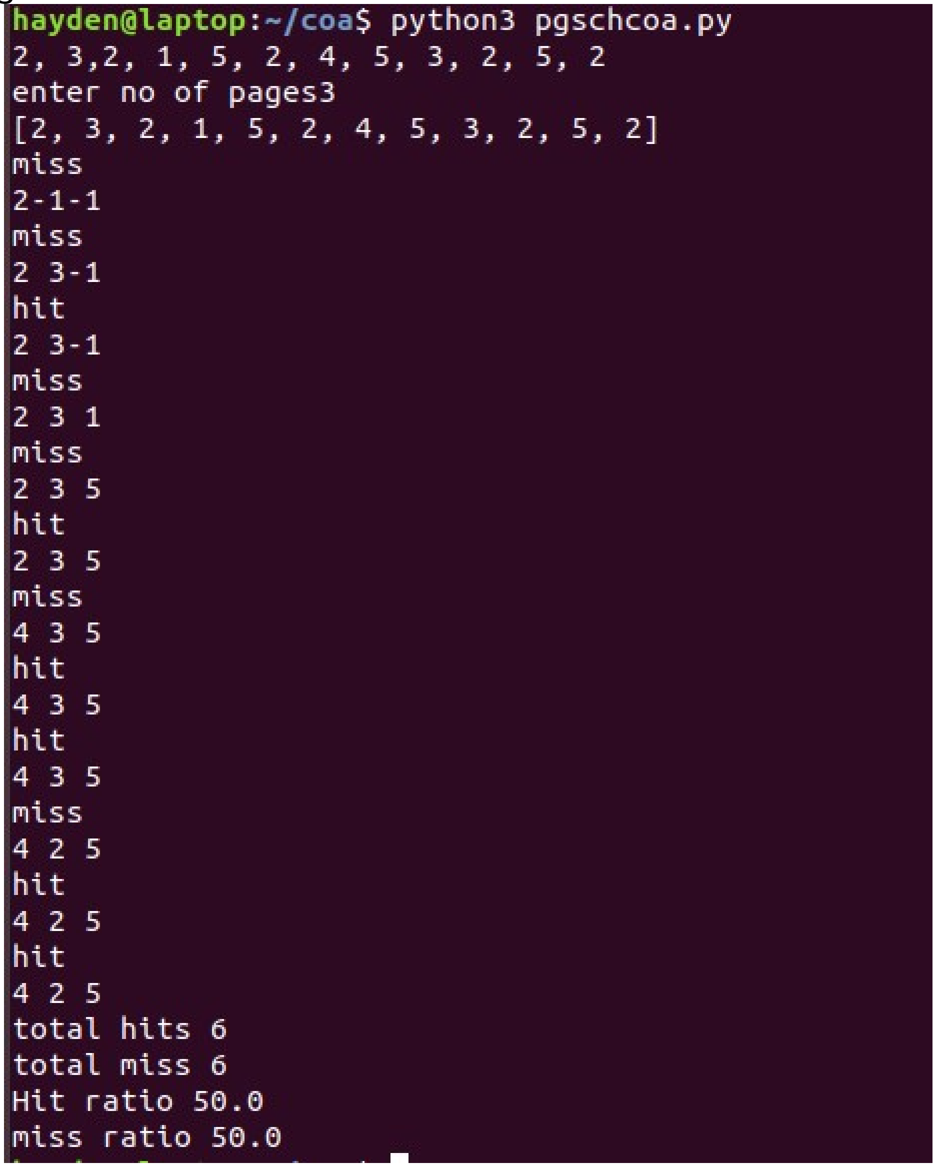
	take a lot of time.
SOFTWARE USED	C/C++/Java
STEPS TO EXECUTE THE PROGRAM	1.Asktheusertoentertheframesize.(ex:takeit3) 2.Lethimenterthenumberofpages. 3. Asktheusertoenterthepagenumbers(referencestring). 4. Initiallythereoccursthree(sameasyourframesize)pagefaultswhilefilling theframe. 5. Afterthatwhentheframeisfull,thepageisreplaceddependingon the specific page replacementalgorithm. 6. Whenever the same page appears in the frame ,a hit occurs. 7.Displayineachclockcyclethecontentsoftheframe.iethepage numbersandshowwhetheritisahitoramiss. 8.Calculatethetotalno.ofhits.missesandthehitratio(no.ofhits/ totalnumber of pages entered) and miss ratio or fault ratio (no.of misses/total number of pages entered).
CODE	

```
def f(nl,l):#returns the page which came first the input list(FCFS)
    fmin=1000
    for i in nl:
        if l.index(i)<fmin:
            fmin=l.index(i)
    return(l[fmin])



def smal(cl,l,fl):#returns the page in the cache list that has to be replaced
    ind=0
    val=0
    nl=[]#if and page doesnt exist in the future it gets appended in this list
    for i in cl:

        try:
            if(l.index(i)>ind):
                ind=l.index(i)

        except:
            nl.append(i)


    if len(nl)==0:#if all page exist in the future
        val=l[ind]
        return(val)
    else:#if one or more page exist in the future
        return f(nl,fl)
```

```
sl=list(map(int,input().split()))#pages input list
# sl=[2, 3,2, 1, 5, 2, 4, 5, 3, 2, 5, 2]
slt=sl#copy of the pages list
cl=[]#frame list(cache list)
hm=[]#hit and miss appends h and m respectively

frm=int(input('enter no of pages'))-1#the frame size
print(slt)
for i in slt:#goes through every page
    ask=input('y/n').lower().strip()#ask the user if they want to continue
    if ask=='n':
        break
    print(i)
    if len(cl)<=frm: #if the cache list is not full
        if i not in cl:
            cl.append(i)
            print('miss')
            hm.append('m')
        elif i in cl:
            print('hit')
            hm.append('h')

    else:#if the cache list is full
        if i in cl:
            print('hit')
            hm.append('h')
        if i not in cl:
            print('miss')
            hm.append('m')
            r=smal(cl,slt,sl)
            cl[cl.index(r)]=i

    if len(cl)==frm+1:
        print(*cl)
    else:
        t=frm+1-len(cl)
        print(*cl,end="")
        for i in range(t):
            print('-1',end="")
        print()


    slt=slt[1:]


print('total hits',hm.count('h'))
print('total miss',hm.count('m'))
print('Hit ratio',(hm.count('h')/len(hm)*100))
print('miss ratio',(hm.count('m')/len(hm)*100))
```

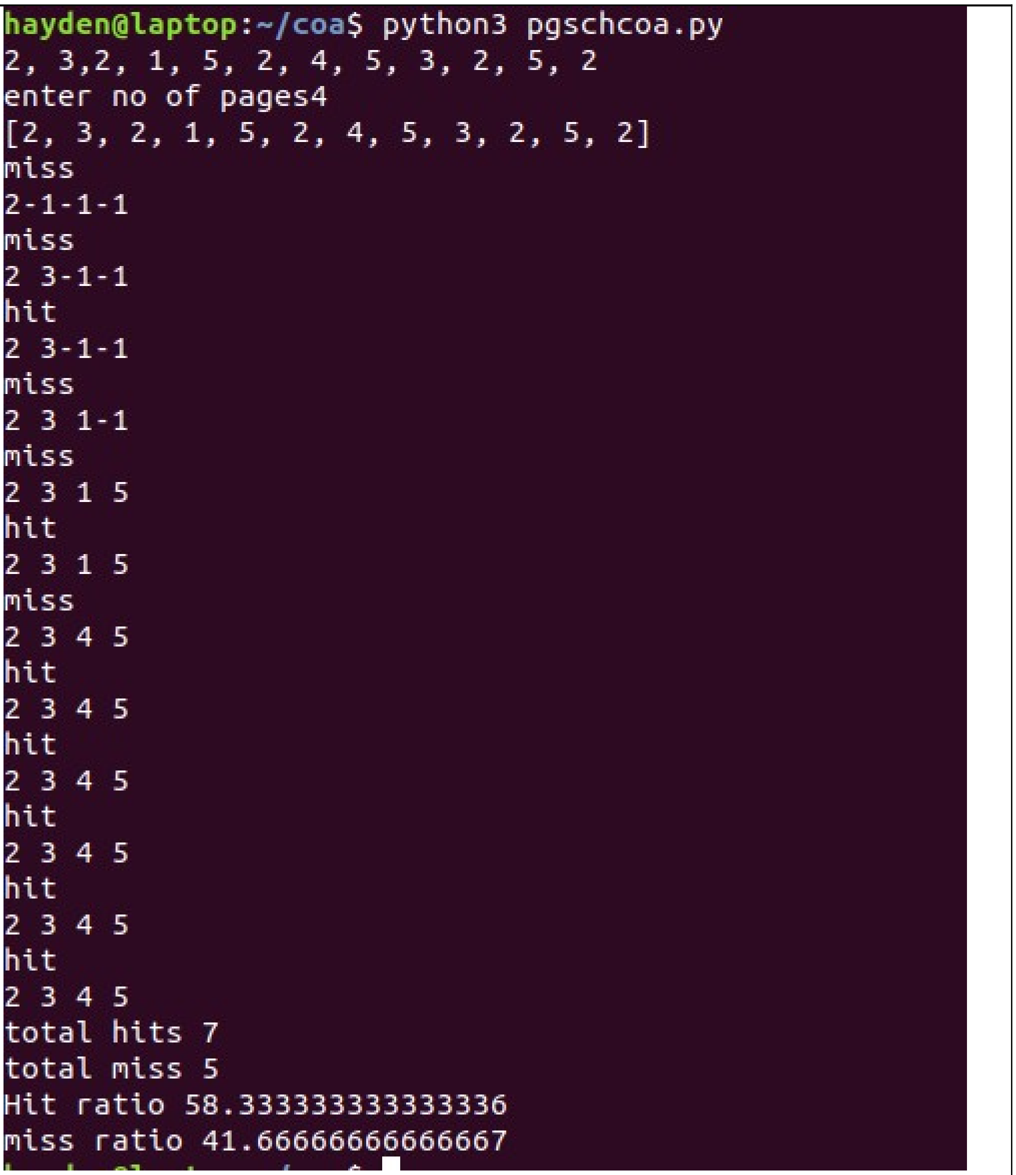
Output:

Pages:3



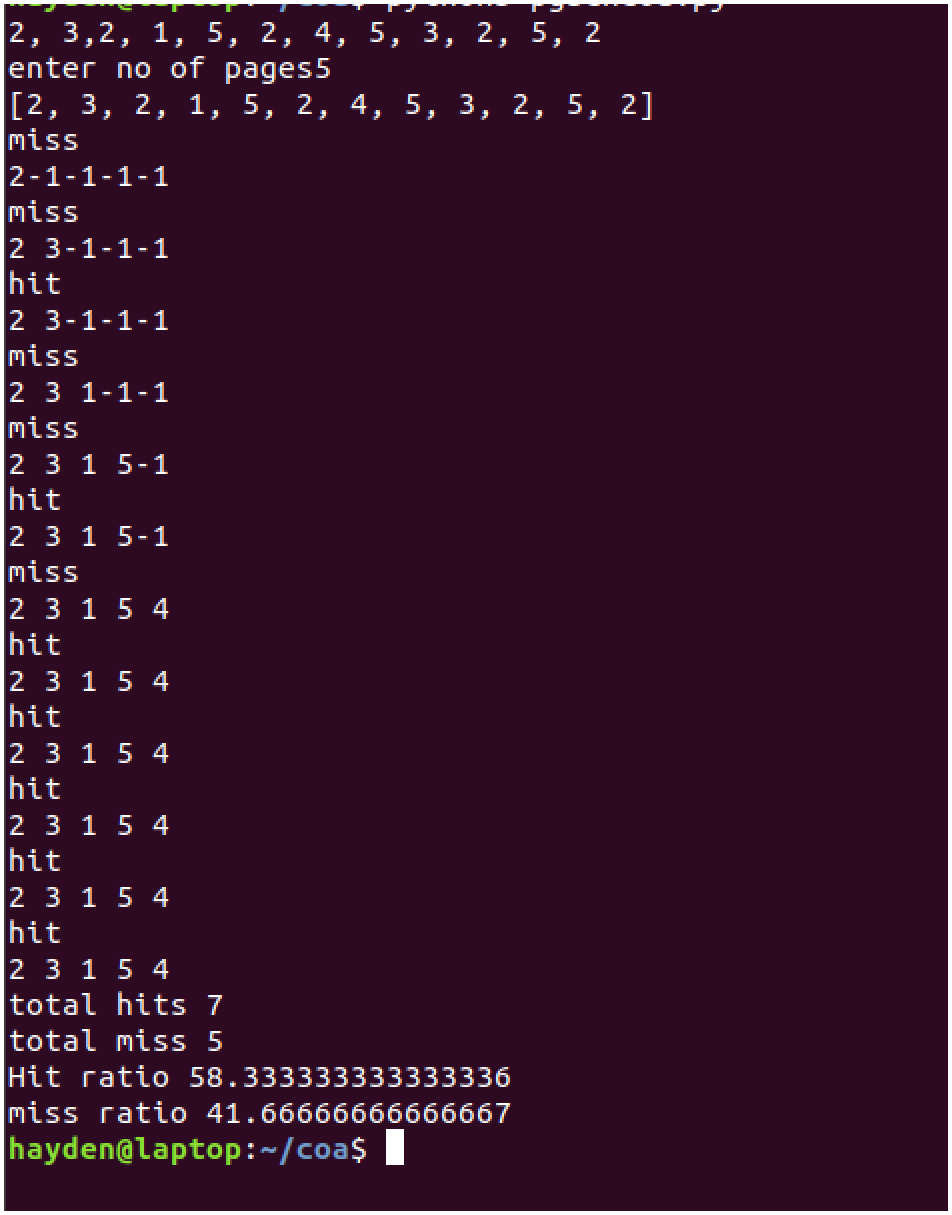
```
hayden@laptop:~/coa$ python3 pgschcoa.py
2, 3,2, 1, 5, 2, 4, 5, 3, 2, 5, 2
enter no of pages3
[2, 3, 2, 1, 5, 2, 4, 5, 3, 2, 5, 2]
miss
2-1-1
miss
2 3-1
hit
2 3-1
miss
2 3 1
miss
2 3 5
hit
2 3 5
miss
4 3 5
hit
4 3 5
hit
4 3 5
miss
4 2 5
hit
4 2 5
hit
4 2 5
total hits 6
total miss 6
Hit ratio 50.0
miss ratio 50.0
```

pages 4:



```
hayden@laptop:~/coa$ python3 pgschcoa.py
2, 3,2, 1, 5, 2, 4, 5, 3, 2, 5, 2
enter no of pages4
[2, 3, 2, 1, 5, 2, 4, 5, 3, 2, 5, 2]
miss
2-1-1-1
miss
2 3-1-1
hit
2 3-1-1
miss
2 3 1-1
miss
2 3 1 5
hit
2 3 1 5
miss
2 3 4 5
hit
2 3 4 5
hit
2 3 4 5
hit
2 3 4 5
hit
2 3 4 5
hit
2 3 4 5
total hits 7
total miss 5
Hit ratio 58.333333333333336
miss ratio 41.66666666666667
```

page 5:



```
2, 3,2, 1, 5, 2, 4, 5, 3, 2, 5, 2
enter no of pages5
[2, 3, 2, 1, 5, 2, 4, 5, 3, 2, 5, 2]
miss
2-1-1-1-1
miss
2 3-1-1-1
hit
2 3-1-1-1
miss
2 3 1-1-1
miss
2 3 1 5-1
hit
2 3 1 5-1
miss
2 3 1 5 4
hit
2 3 1 5 4
hit
2 3 1 5 4
hit
2 3 1 5 4
hit
2 3 1 5 4
hit
2 3 1 5 4
total hits 7
total miss 5
Hit ratio 58.333333333333336
miss ratio 41.66666666666667
hayden@laptop:~/coa$ 
```

CONCLUSION	We have sucessfully understood and implemented optimal page replacement algorithm
REFERENCES	1. WilliamStallings,“ComputerOrganizationandArchitecture:DesigningforPerformance”,PearsonPublication,10thEdition,2013 2. B.Govindarajulu,“ComputerArchitectureandOrganization:Design PrinciplesandApplications”,SecondEdition,McGrawHill(India)