

# OS LAB 5

Ridhima Kohli  
B19CSE071

# How to run ?

Type the following commands on terminal

```
g++ lab5.cpp -o l5
```

```
./l5
```

# Inputs

Enter number of processes

```
Enter number of process k
```

```
3
```

```
Enter Virtual address space : max number of pages :
```

```
8
```

```
Enter Physical address space : number of frames :
```

```
6
```

```
Enter size of TLB(should be less than number of frames):
```

```
3
```

Input max number of pages: Virtual address space :  $m$

Input number of frames : Physical address space:  $f$

Input size of TLB ( $<f$ )  $s$

As shown in the screenshot

Page request of page p from reference string

Search for p in  
TLB

Found  
in TLB

Not  
found  
in TLB

Invalid in  
Page Table

Check Page  
Table

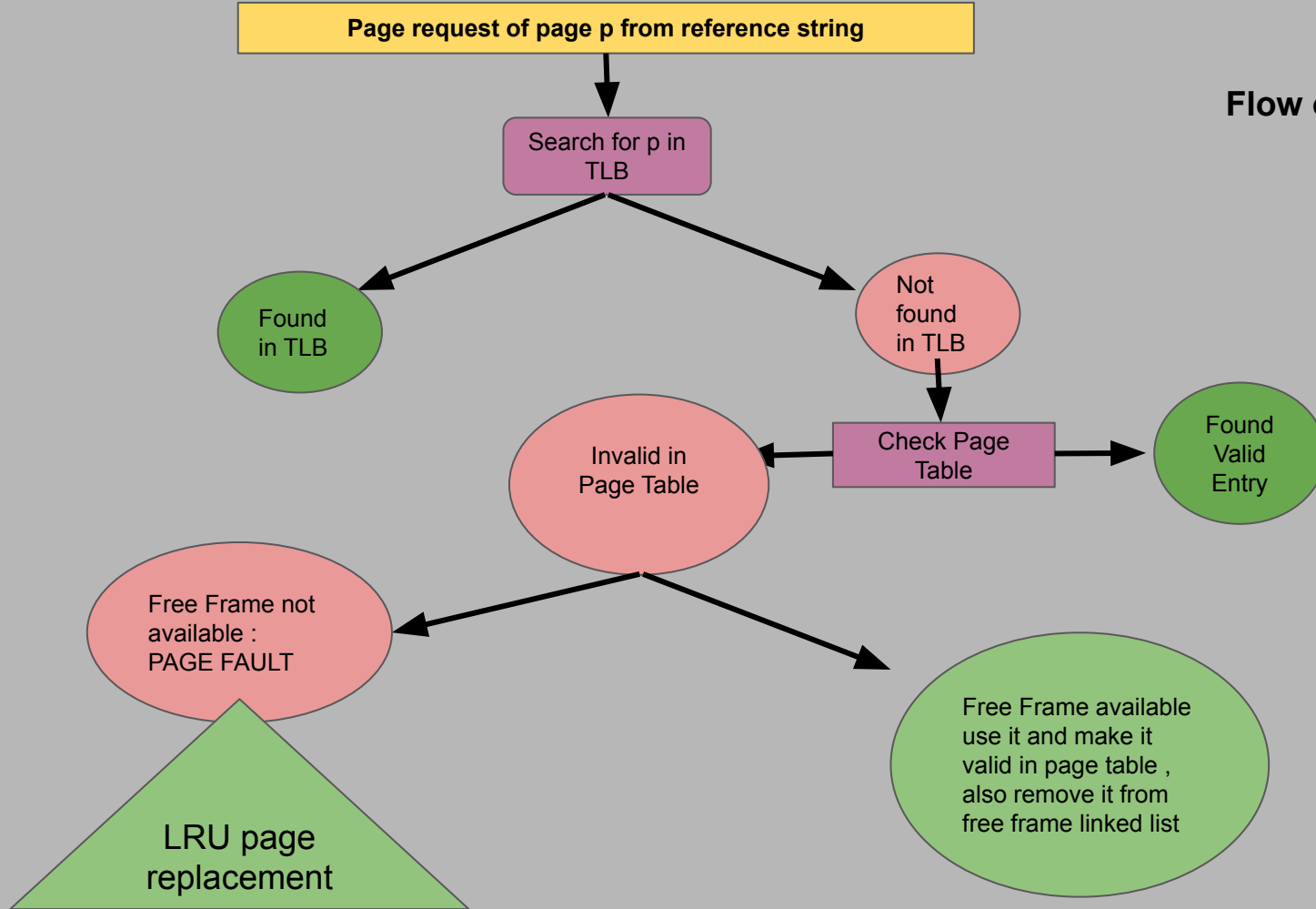
Found  
Valid  
Entry

Free Frame not  
available :  
PAGE FAULT

LRU page  
replacement

Free Frame available  
use it and make it  
valid in page table ,  
also remove it from  
free frame linked list

Flow of program



Component	Data Structure used
TLB	Vector of pairs - {page number , frame number}
Page Table	Vector of pairs - {frame number,validity bit}
Free Frame list	Linked List
Frame-Page link record	Struct FP

The code contains comments to understand the flow and working

# Demand Paging Implementation Correctness

Frames are being brought only when they are being used. Initially all of them are invalid and are present in free frame linked list

During random TLB assignment as mentioned in question , the assigned frames are removed and made valid in page table and linked with corresponding page number.

On subsequent TLB miss , the pages are being linked to pages as per the algorithm diagram shown in previous slides

Hence , frames are linked on demand and demand paging is implemented.

```

PS C:\Users\LENOVO\Desktop\OS_BAB7_713>
Enter number of process k
3
Enter Virtual address space : max number of pages :
8
Enter Physical address space : number of frames :
6
Enter size of TLB(should be less than number of frames):
3
Current process is: 1
Current process requires 6 pages
Reference String Generated : 7 1 4 6 7 1 0 0 2 1 5 4
Free Frames : 3 4 5
TLB : page - frame
2 - 0
3 - 1
6 - 2

Page Table : Page number - Validity (Only frames chosen for TLB are valid)
0 - 1
1 - 1
2 - 1
3 - 0
4 - 0
5 - 0
Starting Process number : 0
Process : 1 for page reference 7 , TLB miss -> Now check page_table
TLB Miss with no page fault , free frame is available
Used the new frame 3
added 7 Current state of LRU:
7
Process : 1 for page reference 1 , TLB miss -> Now check page_table
TLB Miss with no page fault , free frame is available

```

Output on terminal

The complete output is shown in result.txt in the submitted folder