

Ex. No.: 11c)

Date: 23/4/25

### Optimal

Aim:

To write a c program to implement Optimal page replacement algorithm.

#### ALGORITHM:

1. Start the process
2. Declare the size
3. Get the number of pages to be inserted
4. Get the value
5. Declare counter and stack
6. Select the least frequently used page by counter value
7. Stack them according to the selection.
8. Display the values
9. Stop the process

#### PROGRAM:

```
#include <stdio.h>
```

```
int search (int key, int frame[], int size) {  
    for (int i = 0; i < size; i++)  
        if (frame[i] == key) return 1;  
    return 0;  
}
```

```
int predict (int ref[], int frame[], int n, int index,  
             int size) {  
    int farthest = index res = -1;  
    for (int i = 0; i < size; i++) {  
        int j;
```



```

for (j = index; j < n; j++) {
    if (frame[i] == ref[j]) {
        if (j > farthest) {
            farthest = j;
            res = j;
        }
        break;
    }
}
if (j == n) return i;
return (res == -1) ? 0 : res;
}

```

```

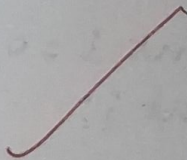
}
void printFrame (int item, int frame[], int size,
                int max) {
    printf("%d\t", item);
    for (int i = 0; i < max; i++)
        printf("%d\t", (i < size) ? frame[i] : -1);
    printf("\n");
}

```

```

}
void optimalPage (int ref[], int n, int max) {
    int frame[max], filled = 0, hits = 0;
    printf("Page\t");
    for (int i = 1; i <= max; i++)
        printf("Frame %d\t", i);
    printf("\n");
}

```





```
for (int i = 0; i < n; i++) {
```

```
    if (search(ref[i], frame, filled))
```

```
        hits++;
```

```
    else if (filled < max)
```

```
        frame[filled++] = ref[i];
```

```
    else
```

```
        frame(predict ref, frame, n, i + 1, max) =
```

```
            ref[i];
```

```
    printFrame(ref[i], frame, filled, max);
```

```
}
```

```
printf("\n Hits: %d \n Misses: %d \n, hits, n-hits);
```

```
}
```

```
int main()
```

```
{
```

```
    int ref[] = {7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1};
```

```
    int n = size of (ref) / size of (ref[0]);
```

```
    optimalPage(ref, n, 3);
```

```
    return 0;
```

```
}
```

OUTPUT:

stream

7	7	-1	-1
0	7	0	-1
1	7	0	1
2	2	0	1
0	2	0	1
3	2	0	3

0	2	0	3
4	2	4	3
2	2	4	3
3	2	4	3
0	2	0	3
3	2	0	3
2	2	0	3
1	2	0	1
2	2	0	1
0	2	0	1
1	2	0	1
7	7	0	1
0	7	0	1
1	7	0	1

Hits : 11

Misses : 9.



Output:

Result: Using C the optimal page replacement algorithm is implemented.

Q. 11