Experiment-33:Construct a C program to simulate the optimal paging technique of memory management

Aim:
To simulate the Optimal Paging technique of memory management in C.

Procedure:

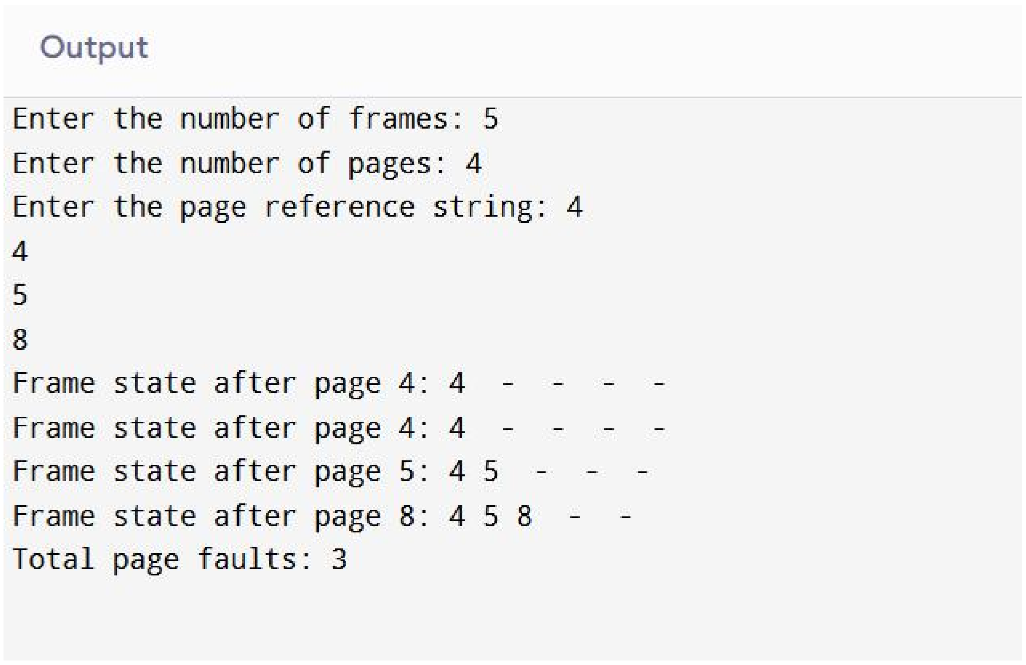
1. Take the number of pages and the number of frames as input.
2. Simulate the Optimal paging algorithm by looking ahead in the page reference string and replacing the page that will not be used for the longest period of time.
3. Keep track of page faults and display the results.

C Program:

```
#include <stdio.h>


int main() {

   int frames, pages, page_faults = 0;

   printf("Enter the number of frames: ");

   scanf("%d", &frames);

   printf("Enter the number of pages: ");

   scanf("%d", &pages);


   int page_sequence[pages], frame[frames];

   for (int i = 0; i < frames; i++) {

      frame[i] = -1;

   }


   printf("Enter the page reference string: ");

   for (int i = 0; i < pages; i++) {

      scanf("%d", &page_sequence[i]);

   }
```

```
for (int i = 0; i < pages; i++) {
    int page_found = 0, farthest = -1, replace_index = -1;
    for (int j = 0; j < frames; j++) {
        if (frame[j] == page_sequence[i]) {
            page_found = 1;
            break;
        }
    }

    if (!page_found) {
        if (page_faults < frames) {
            frame[page_faults] = page_sequence[i];
            page_faults++;
        } else {
            for (int j = 0; j < frames; j++) {
                int next_use = -1;
                for (int k = i + 1; k < pages; k++) {
                    if (frame[j] == page_sequence[k]) {
                        next_use = k;
                        break;
                    }
                }

                if (next_use == -1) {
                    replace_index = j;
                    break;
                    }

                    if (next_use > farthest) {
                        farthest = next_use;
                        replace_index = j;
                    }
                }
                frame[replace_index] = page_sequence[i];
                page_faults++;
            }
        }

        printf("Frame state after page %d: ", page_sequence[i]);
        for (int j = 0; j < frames; j++) {
            if (frame[j] != -1) {
                printf("%d ", frame[j]);
            } else {
                printf(" - ");
            }
        }
        printf("\n");
    }

    printf("Total page faults: %d\n", page_faults);
    return 0;
}
```

Output:



```
Enter the number of frames: 5
Enter the number of pages: 4
Enter the page reference string: 4
4
5
8
Frame state after page 4: 4  -   -   -   -
Frame state after page 4: 4  -   -   -   -
Frame state after page 5: 4 5  -   -   -
Frame state after page 8: 4 5 8  -   -
Total page faults: 3
```