Optimal

```
#include <stdio.h>

int main() {
   int pages[30], frame[10], n, f, i, j, k, pos, max, flag1, flag2, fault = 0;

   printf("Enter number of pages: ");
   scanf("%d", &n);
   printf("Enter the page reference string:\n");
   for(i = 0; i < n; i++)
      scanf("%d", &pages[i]);

   printf("Enter number of frames (min 3): ");
   scanf("%d", &f);

   for(i = 0; i < f; i++)
      frame[i] = -1;

   printf("\nPage\tFrames\t\tPage Fault\n");
   for(i = 0; i < n; i++) {
      flag1 = flag2 = 0;

      for(j = 0; j < f; j++) {
         if(frame[j] == pages[i]) {
            flag1 = flag2 = 1;
            break;
         }
      }

      if(flag1 == 0) {
         for(j = 0; j < f; j++) {
```

```
        if(frame[j] == -1) {
            frame[j] = pages[i];
            fault++;
            flag2 = 1;
            break;
        }
    }
}

if(flag2 == 0) {
    int next[10];
    for(j = 0; j < f; j++) {
        next[j] = -1;
        for(k = i + 1; k < n; k++) {
            if(frame[j] == pages[k]) {
                next[j] = k;
                break;
            }
        }
    }

    pos = 0;
    max = next[0];
    for(j = 1; j < f; j++) {
        if(next[j] == -1) {
            pos = j;
            break;
        }
        if(next[j] > max) {
            max = next[j];
            pos = j;
```

```
            }
         }
         frame[pos] = pages[i];
         fault++;
      }

      printf("%d\t", pages[i]);
      for(j = 0; j < f; j++) {
         if(frame[j] != -1)
            printf("%d ", frame[j]);
         else
            printf("- ");
      }
      if(flag1 == 0)
         printf("\tPage Fault %d", fault);
      printf("\n");
   }

   printf("\nTotal Page Faults: %d\n", fault);
   return 0;
}
```

```
Enter number of pages: 10
Enter the page reference string:
7 0 1 2 0 3 0 4 2 3
Enter number of frames: 3
Page    Frames        Page Fault
7       7 - -         Page Fault 1
0       7 0 -         Page Fault 2
1       7 0 1         Page Fault 3
2       2 0 1         Page Fault 4
```

0 2 0 1

3 2 0 3 Page Fault 5

0 2 0 3

4 4 0 3 Page Fault 6

2 4 0 2 Page Fault 7

3 4 0 2

Total Page Faults: 7

⚙️ 3️⃣ **OPTIMAL (Future Knowledge)**

🧠 **Logic:**

- Replace the page that will **not be used for the longest time in the future**.
- This gives the *lowest possible page faults*.

📋 **Step-by-Step Table:**

Step	Page	Frames	Page Fault?	Explanation
1	7	7 - -	✅	7 inserted
2	0	7 0 -	✅	0 inserted
3	1	7 0 1	✅	1 inserted
4	2	2 0 1	✅	7 replaced (not used again soon)
5	0	2 0 1	❌	0 already present
6	3	2 0 3	✅	1 replaced (next used farthest in future)
7	0	2 0 3	❌	0 already present
8	4	4 0 3	✅	2 replaced (not used again soon)
9	2	4 0 2	✅	3 replaced (not used again soon)
10	3	4 0 2	❌	3 not used again (no replacement)

✅ **Total Page Faults = 7**

🧩 **Explanation:**
The Optimal algorithm looks *ahead* and replaces the page that won't be needed for the longest time. That's why it's the **best performing algorithm** — but it's **impractical** in real systems (you can't predict the future).