Slip 22

Program 1: Orphan Process Simulation

```
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>

int main() {
   pid_t pid = fork();

   if (pid == 0) {
      // Child process
      sleep(5);  // Child sleeps to ensure parent terminates first
      printf("Orphan Child Process: PID = %d, Parent PID = %d\n", getpid(), getppid());
   }
   else if (pid > 0) {
      // Parent process
      printf("Parent Process: PID = %d, Child PID = %d\n", getpid(), pid);
      printf("Parent Process terminates...\n");
   }
   else {
      // Fork failed
      printf("Fork failed!\n");
   }

   return 0;
}
```

Program 2: Optimal Page Replacement Algorithm

```
#include <stdio.h>
#define MAX 20

int frames[MAX], ref[MAX], mem[MAX][MAX], faults = 0, m, n;

void accept() {
   printf("Enter number of frames: ");
   scanf("%d", &n);
   printf("Enter number of references: ");
   scanf("%d", &m);
```

```
    printf("Enter reference string:\n");
    for (int i = 0; i < m; i++) {
        printf("[%d] = ", i);
        scanf("%d", &ref[i]);
    }
}

int search(int pno) {
    for (int i = 0; i < n; i++) {
        if (frames[i] == pno) return i;
    }
    return -1;
}

int predict(int current_index) {
    int farthest = current_index, pos = -1;
    for (int i = 0; i < n; i++) {
        int j;
        for (j = current_index; j < m; j++) {
            if (frames[i] == ref[j]) {
                if (j > farthest) {
                    farthest = j;
                    pos = i;
                }
                break;
            }
        }
        if (j == m) return i;
    }
    return (pos == -1) ? 0 : pos;
}

void optimal_page_replacement() {
    for (int i = 0; i < m; i++) {
        if (search(ref[i]) == -1) {
            if (i < n) {
                frames[i] = ref[i];
            } else {
                int pos = predict(i + 1);
```

```
            frames[pos] = ref[i];
          }
          faults++;
        }
        for (int j = 0; j < n; j++) {
          mem[j][i] = frames[j];
        }
    }
}

void disp() {
    printf("\nReference String:\n");
    for (int i = 0; i < m; i++) {
        printf("%3d", ref[i]);
    }
    printf("\n\nFrame Allocation:\n");
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            if (mem[i][j]) {
                printf("%3d", mem[i][j]);
            } else {
                printf("   ");
            }
        }
        printf("\n");
    }
    printf("\nTotal Page Faults: %d\n", faults);
}

int main() {
    accept();
    optimal_page_replacement();
    disp();
    return 0;
}
```