Program 1

```
#include <stdio.h>

#define MAX 20

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


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 get_lfu() {

int min = 9999, min_i = 0;

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

if (count[i] < min) {

min = count[i];

min_i = i;

}

}

return min_i;

}


void lfu() {

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

int k = search(ref[i]);
if (k == -1) {

if (sp < n) {

frames[sp] = ref[i];

count[sp] = 1;

sp++;

} else {

int pos = get_lfu();

frames[pos] = ref[i];

count[pos] = 1;

}

faults++;

} else {

count[k]++;

}

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();

lfu();

disp();

return 0;

}
```

Program 2

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

#include <sys/stat.h>

#include <fcntl.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <dirent.h>

#include <unistd.h>


void make_toks(char *s, char *tok[]) {

    int i = 0;

    char *p;

    p = strtok(s, " ");

    while (p != NULL) {

        tok[i++] = p;

        p = strtok(NULL, " ");

    }

    tok[i] = NULL;

}


void list(char *dn, char op) {

    DIR *dp;

    struct dirent *entry;

    int dc = 0, fc = 0;


    dp = opendir(dn);

    if (dp == NULL) {

        printf("Dir %s not found.\n", dn);

        return;

    }
    switch (op) {
        case 'f':
            while ((entry = readdir(dp)) != NULL) {
                if (entry->d_type == DT_REG)
                    printf("%s\n", entry->d_name);
            }
            break;
        case 'n':
            while ((entry = readdir(dp)) != NULL) {
                if (entry->d_type == DT_DIR) dc++;
                if (entry->d_type == DT_REG) fc++;
            }
            printf("%d Dir(s)\t%d File(s)\n", dc, fc);
            break;
        case 'i':
            while ((entry = readdir(dp)) != NULL) {
                if (entry->d_type == DT_REG)
                    printf("%s\t%lu\n", entry->d_name, entry->d_fileno);
            }
            break;
    }

    closedir(dp);
}

int main() {
    char buff[80], *args[10];
    int pid;

    while (1) {
        printf("myshell$ ");
        fflush(stdin);
        fgets(buff, 80, stdin);
        buff[strlen(buff) - 1] = '\0';
        make_toks(buff, args);

        if (strcmp(args[0], "list") == 0) {
            list(args[2], args[1][0]);
        } else {
            pid = fork();
            if (pid > 0) {
                wait(NULL);
            } else {
                if (execvp(args[0], args) == -1) {
                    printf("Bad command.\n");
                }
            }
        }
    }
    return 0;
}
```