```c
//NOTO: il problema può essere risolto usando una pila implementata come lista semplice, senza aumentare il costo computazionale.

#include <stdio.h>
#include <stdlib.h>

// struct per memorizzare ogni riga
typedef struct
{
    int n;
    float f;
    char s[6 + 1];     // ogni stringa deve avere spazio per \0
    struct data *succ; // Puntatore all'elemento successivo della lista
    struct data *prec;
} data;

int main()
{
    FILE *fp;                // Puntatore al file prima di input, poi di outpu
    data *head, *tail, *tmp; // HEAD: inzio lista, TAIL: fine lista, TMP: variabile di supporto
    // Inizializzazione lista vuota
    head = NULL;
    tail = NULL;

    // Apro file input
    fp = fopen("data_es2_input", "r");
    if (fp == NULL)
        return (EXIT_FAILURE);

    // Alloco il primo elemento della lista
    tmp = (data *)malloc(sizeof(data));
    tmp->prec = NULL;
    tmp->succ = NULL;
    head = tmp;
    tail = tmp;

    // Inserimento in fondo alla lista fino a quando la lista non è vuota
    while (fscanf(fp, " %d %f %6s", &(tmp->n), &(tmp->f), tmp->s) != EOF)
    {
        tmp = (data *)malloc(sizeof(data));
        tmp->prec = tail;
        tmp->succ = NULL;
        tail->succ = tmp;
        tail = tmp;
    }
    // il ciclo while viene eseguito una volta di troppo, quindi libero tmp aggiornando la coda
    // Ho preferito usare questo espediente in modo da poter usare fscanf direttamente sull'elemento
    // della lista e non su una variabile d'appoggio. Questo permette di non dover copiare ogni volta la struct
    tail = tmp->prec;
```

```c
    free(tmp);

    // Chiudo il file di input e apro il file di output
    fclose(fp);
    fp = fopen("data_es2_output", "w");
    if (fp == NULL)
        return (EXIT_FAILURE);
    // Visita dalla coda. Scrivo su file e libero memoria. Poi termino
    while (tail != NULL)
    {
        fprintf(fp, "%d %f %6s\n", tail->n, tail->f, tail->s);
        tmp = tail->prec;
        free(tail);
        tail = tmp;
    }
    fclose(fp);
    return 0;
}

// Implementazione lista semplice
// int main()
// {
//      data *tmp, *head = NULL;
//     FILE *fp;
//     fp = fopen("data_es2_input", "r");
//     tmp = (data *)malloc(sizeof( data));
//     while (fscanf(fp, " %d %f %6s", &(tmp->n), &(tmp->f), tmp->s) != EOF)
//     {
//         tmp->next = head;
//         head = tmp;
//         tmp = (data *)malloc(sizeof(data));
//     }
//     free(tmp);
//     fclose(fp);
//     fp = fopen("data_es2_output", "w");
//     while (head != NULL)
//     {
//         fprintf(fp, " %d %f %6s\n", head->n, head->f, head->s);
//         tmp = head;
//         free(head);
//         head = tmp->next;
//     }
//     fclose(fp);
//     free(head);
//     return 0;
// }

```