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

struct list {
    int data;
    struct list *next;
};

void print_list(struct list *head) {

    // empty list check
    if (head == NULL) {
        printf("empty list!\n");
        return;
    }

    // guaranteed non-empty list
    struct list *cursor = head;

    while (cursor != NULL)
    {
        printf("cursor element is : %c\n", cursor->data);
        cursor = cursor->next;
    }
    /*
    if (cursor == NULL)
        return;

    printf("cursor element is : %c\n", cursor->data);
    cursor = cursor->next;
    if (cursor == NULL)
        return;

    printf("cursor element is : %c\n", cursor->data);
    cursor = cursor->next;
    */
}

void pfree(void *address) {

    struct list *tmp = (struct list*)address;
    printf("freeing %p %c\n", address, tmp->data);
    free(address);
}
```

```c
void list_free(struct list *head)
{
    if (head == NULL)
        return;

    // at this point 1+ elements
    struct list *cursor = head;

    if (cursor->next == NULL) {
        // element 1 only case
        pfree(head);
        return;
    }

    // at this point 2+ cases
    struct list *tmp;
    while (cursor != NULL)
    {
        tmp = cursor->next;
        pfree(cursor);
        cursor = tmp;
    }
}

int main() {
    struct list *cursor;

    cursor = (struct list*)malloc(sizeof(struct list));

    cursor->data = 'b';
    cursor->next = (struct list*)malloc(sizeof(struct
     list));

    struct list *tmp = cursor->next;
    tmp->data = 'c';
    tmp->next = (struct list *)malloc(sizeof(struct list));

    tmp->next->data = 'd';
    tmp->next->next = NULL;

    print_list(cursor);

    list_free(cursor->next);

    return 0;
```