Rezolvari Tutoriat 6

Programarea Calculatoarelor

07.12.2018

1

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

struct nod {
    int val;
    struct nod *urm;
};

struct nod *prim, *ultim;

void adaugaNod(int x) {
    struct nod *noulNod = (struct nod*)malloc(sizeof(struct
nod));
    noulNod->val = x;
    noulNod->urm = NULL;
    if (prim == NULL) {
        prim = noulNod;
        ultim = noulNod;
    }
    else {
        ultim->urm = noulNod;
        ultim = noulNod;
    }
}

void sterge() {
    struct nod* p = prim;
    prim = prim->urm;
    if (prim == NULL) ultim = NULL;
    free(p);
}

void afisare(FILE* out) {
    struct nod *p = prim;
    while (p != NULL) {
        fprintf(out, "%d ", p->val);
        p = p->urm;
    }
}

int main() {
    FILE *f, *g;
    f = fopen("/Users/alexchirea/Desktop/lista.in", "r");
    g = fopen("/Users/alexchirea/Desktop/lista.out", "w");
    prim = ultim = NULL; // SAU prim = ultim = 0;
    int k, i, x;
```

Alexandru Chirea
ac.chirea@gmail.com

```
    fscanf(f, "%d", &k);
    while (fscanf(f, "%d", &x) != EOF) {
        adaugaNod(x);
    }
    for(i=0; i<k; i++) {
        if(prim != NULL) {
            adaugaNod(prim->val);
            sterge();
        }
    }
    afisare(g);
    fclose(f);
    fclose(g);
    return 0;
}
```

2

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

typedef struct node {
    int val;
    struct node *urm;
} nod;

typedef struct list {
    nod *primulElement;
} lista;

void adaugaNod(int x, lista *start) {
    nod *noulNod = (nod*)malloc(sizeof(nod));
    noulNod->val = x;
    noulNod->urm = NULL;
    if (start->primulElement == NULL) {
        start->primulElement = noulNod;
    }
    else {
        nod* ultim = start->primulElement;
        while (ultim->urm != NULL) {
            ultim = ultim->urm;
        }
        ultim->urm = noulNod;
    }
}

int sterge(lista* start) {
    nod* p = start->primulElement;
    start->primulElement = start->primulElement->urm;
    int x = p->val;
    free(p);
    return x;
```

Alexandru Chirea
ac.chirea@gmail.com

```
}

void afisare(lista *start) {
    nod *p = start->primulElement;
    while (p != NULL) {
        printf("%d ", p->val);
        p = p->urm;
    }
}

int main() {
    lista pare = {NULL};
    lista impare = {NULL};
    int i, n, x;
    scanf("%d", &n);
    for (i=0; i<n; i++) {
        scanf("%d", &x);
        if (x % 2) adaugaNod(x, &impare);
        else adaugaNod(x, &pare);
    }
    afisare(&pare);
    printf("\n");
    afisare(&impare);
    printf("\n");
    while (impare.primulElement != NULL) {
        adaugaNod(sterge(&impare), &pare);
    }
    afisare(&pare);
    printf("\n");
    return 0;
}
```

3

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

typedef struct node {
    int val;
    struct node *urm;
} nod;

typedef struct list {
    nod *primulElement;
} lista;

void adaugaNod(int x, lista *start) {
    nod *noulNod = (nod*)malloc(sizeof(nod));
    noulNod->val = x;
    noulNod->urm = NULL;
    if (start->primulElement == NULL) {
        start->primulElement = noulNod;
```

```
    }
    else {
        nod* ultim = start->primulElement;
        while (ultim->urm != NULL) {
            ultim = ultim->urm;
        }
        ultim->urm = noulNod;
    }
}

void sterge(int x, lista* start) {
    nod *p = start->primulElement;
    while (p->urm != NULL) {
        nod *aux;
        if (p->val == x && p->urm->urm != NULL) {
            //primul nod din lista
            if (start->primulElement == p) {
                aux = start->primulElement;
                start->primulElement =
start->primulElement->urm;
                free(aux);
                return;
            }
            //oricare alt nod
            else {
                aux = start->primulElement;
                while (aux->urm->val != p->val) {
                    aux = aux->urm;
                }
                aux->urm = aux->urm->urm;
                free(p);
                return;
            }
        }
        //ultimul nod din lista
        if (p->urm->urm == NULL && p->urm->val == x) {
            aux = p->urm;
            p->urm = NULL;
            free(aux);
            return;
        }
        p = p->urm;
    }
}

void afisare(lista *start) {
    nod *p = start->primulElement;
    while (p != NULL) {
        printf("%d ", p->val);
        p = p->urm;
```

Alexandru Chirea
ac.chirea@gmail.com

```
    }
}

int main() {
    lista Lista = {NULL};
    int x, k, n, i;
    scanf("%d", &n);
    for (i=0; i<n; i++) {
        scanf("%d", &x);
        adaugaNod(x, &Lista);
    }
    scanf("%d", &k);
    afisare(&Lista);
    sterge(k, &Lista);
    printf("\n");
    afisare(&Lista);
    return 0;
}
```

4

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

typedef struct node {
    int val;
    struct node *urm;
} nod;

typedef struct list {
    nod *primulElement;
} lista;

void adaugaNod(int x, lista *start) {
    nod *noulNod = (nod*)malloc(sizeof(nod));
    noulNod->val = x;
    noulNod->urm = NULL;
    if (start->primulElement == NULL) {
        start->primulElement = noulNod;
    }
    else {
        nod* ultim = start->primulElement;
        while (ultim->urm != NULL) {
            ultim = ultim->urm;
        }
        ultim->urm = noulNod;
    }
}

void pare(lista *start) {
    nod *p = start->primulElement;
    while (p != NULL) {
```

```
        if (p->val % 2 == 0) {
            nod *x = (nod*)malloc(sizeof(nod));
            x->urm = p->urm;
            x->val = p->val/2;
            p->urm = x;
            p = p->urm->urm;
        }
        else {
            p = p->urm;
        }
    }
}

void afisare(lista *start) {
    nod *p = start->primulElement;
    while (p != NULL) {
        printf("%d ", p->val);
        p = p->urm;
    }
}

int main() {
    lista Lista = {NULL};
    int x, n, i;
    scanf("%d", &n);
    for (i=0; i<n; i++) {
        scanf("%d", &x);
        adaugaNod(x, &Lista);
    }
    afisare(&Lista);
    pare(&Lista);
    printf("\n");
    afisare(&Lista);
    return 0;
}
```