Minta második ZH

Czirkos Zoltán · 2017.11.14.
Minta második ZH, előző évek feladataiból összeollózva.

Gondolatjelek

Írj programot, amely a szabványos bemeneten érkező szövegből kitörli a gondolatjelek közötti részeket – olyanokat, mint ez –, és úgy írja ki a szabványos kimenetére! A gondolatjelet az különbözteti meg a kötőjeltől, hogy előtte egy szóköz van.

Rajzold meg az ehhez tartozó állapot- és tevékenységtáblát, és abból kiindulva írd meg a programot! *BEUGRÓ:* a programnak állapotgépesnek kell lennie.

Megoldás

	szóköz	**mínusz**	**többi**
szöveg	→lehet	ki: c	ki: c
lehet	–	→gondolat	ki: szóköz, c, →szöveg
gondolat	→vége?	–	–
vége?	–	→szöveg	→gondolat

```
#include <stdio.h>

int main() {
    typedef enum GondolatAll { szoveg, lehet, gondolat, vege } GondolatAll;
    int c;
    GondolatAll allapot = szoveg;

    while ((c = getchar()) != EOF) {
        switch (allapot) {
            case szoveg:
                if (c == ' ') allapot = lehet;
                else putchar(c);
                break;
            case lehet:
                if (c == ' ') putchar(' ');
                else if (c == '-') allapot = gondolat;
                else { putchar(' '); putchar(c); allapot = szoveg; }
                break;
            case gondolat:
                if (c == ' ') allapot = vege;
                break;
            case vege:
                if (c == ' ') /* semmi */;
                else if (c == '-') allapot = szoveg;
                else allapot = gondolat;
                break;
        }
    }

    return 0;
}
```

Pontozás:

- Bemenet sztringbe beolvasva, vagy iterációnként több karakter beolvasása: nincs meg a beugró, nem állapotgépes a megoldás.
- Állapotgép: helyes állapotátmenetek 3p, helyes tevékenységek 1p.
- Program: felsorolt típus 1p, beolvasás 1p, állapotgép kódolása 4p.

Továbbá az általános pontozási irányelvek.

Leghosszabb szó

Írj programot, mely a szabványos bemenetről fájl végéig olvas egy szöveget, a benne található leghosszabb szót megtalálja, és a futás végén a standard kimenetre írja! Egy szó alatt a bemeneti karakterfolyam olyan szakaszát értjük, melyben csak angol betűk vannak. A szavak bármilyen hosszúak lehetnek! (Javasolt saját függvényt írni a sztringhez karakter hozzáfűzéshez és sztring értékadáshoz.)

Megoldás

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

void ertekadas(char **hova, char *mit) {
    free(*hova);
    *hova = malloc(strlen(mit) + 1);
    strcpy(*hova, mit);
}
void hozzafuz(char **str, char mit) {
    char *uj = malloc(strlen(*str) + 2);
    sprintf(uj, "%s%c", *str, mit);
    free(*str);
    *str = uj;
}

int main() {
    char *akt = NULL, *legh = NULL;
    int c;

    ertekadas(&akt, "");
    ertekadas(&legh, "");
    while ((c=getchar()) != EOF) {
        if (isalpha(c))
            hozzafuz(&akt, c);
        else {
            if (strlen(akt) > strlen(legh))
                ertekadas(&legh, akt);
            ertekadas(&akt, "");
        }
    }
    printf("leghosszabb: %s\n", legh);
    free(akt);
```

```
    free(legh);

    return 0;
}
```

Pontozás:

- főprogram: inicializálás 1p, beolvasás 1p, beolvasás ciklusfeltétele 1p, maximumkeresés 1p
- hozzáfűzés: új terület foglalása 1p, régi felszabadítása 1p, pointer visszaadása 1p
- értékadás: felszabadítás 1p, új terület foglalása 1p, pointer visszaadása 1p

Továbbá az általános pontozási irányelvek.

Átszállások

Szöveges fájlokban buszjáratok megállóinak neveit tároljuk, soronként egyet. A programod feladata, hogy láncolt listákba beolvassa a buszjáratok adatait, és megmondja két adott járatról, hogy át lehet-e szállni egyikről a másikra, vagy nem; és ha igen, hol.

a) Írj függvényt, amely paraméterként egy fájlnevet kap, visszatérési értéke pedig egy járat megállóinak listája (a fájlban szereplő sorrendben).

b) Írj függvényt, amely megvizsgál két, paraméterként kapott megállólistát, hogy van-e átszállási lehetőség (azonos megállónév). A függvény visszatérési értéke egy sztring legyen, amely a megálló neve.

c) Egészítsd ki mindezt teljes programmá, amelyben beolvasod a 7E.txt és az M3.txt nevű fájlokat! Ha át lehet szállni, írd ki a megálló nevét, ha nem, akkor pedig írd ki azt! Ne felejtsd el felszabadítani a memóriát, amit foglaltál!

🗁 **Megoldás**

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

typedef struct Megallo {
    char nev[60+1];
    struct Megallo *kov;
} Megallo;

Megallo *vegere(Megallo *lista, char const *ujnev) {
    Megallo *uj = (Megallo*) malloc(sizeof(Megallo));
    strcpy(uj->nev, ujnev);
    uj->kov = NULL;
    if (lista == NULL) {
        lista = uj;
    } else {
        Megallo *utolso;
        for (utolso = lista; utolso->kov != NULL; utolso = utolso->kov)
            ;
        utolso->kov = uj;
    }
    return lista;
}

Megallo *jarat_beolvas(char const *fajlnev) {
    FILE *fp = fopen(fajlnev, "rt");
    if (fp == NULL)
        return NULL;
    char megallo[60+1];
    Megallo *jarat = NULL;
    while (fscanf(fp, " %[^\n]", megallo) != EOF)
        jarat = vegere(jarat, megallo);
    fclose(fp);
    return jarat;
}

void felszabadit(Megallo *lista) {
    while (lista != NULL) {
        Megallo *kov = lista->kov;
        free(lista);
        lista = kov;
    }
}

char const *hol_van_kozos(Megallo *j1, Megallo *j2) {
    Megallo *it1, *it2;
    for (it1 = j1; it1 != NULL; it1 = it1->kov)
        for (it2 = j2; it2 != NULL; it2 = it2->kov)
            if (strcmp(it1->nev, it2->nev) == 0)
                return it1->nev;
    return NULL;
}

int main(void) {
    Megallo *j1, *j2;
    j1 = jarat_beolvas("7E.txt");
    j2 = jarat_beolvas("M3.txt");
    if (j1 == NULL || j2 == NULL) {
        printf("Nem sikerült beolvasni valamelyik járatot\n");
    } else {
        char const *atszallas = hol_van_kozos(j1, j2);
        if (atszallas == NULL)
            printf("Nincs közös megálló\n");
        else
            printf("A közös megálló: %s\n", atszallas);
    }
    felszabadit(j1);
    felszabadit(j2);
    return 0;
}
```

Pontozás:

- típus 1p.
- lista végéhez fűzés 4p.
- járat beolvasása hibaellenőrzéssel 4p.
- n^2 keresés 4p.
- felszabadítás 2p, ha nincs duplikálva a kód.
- főprogram 3p.

Továbbá az általános pontozási irányelvek.