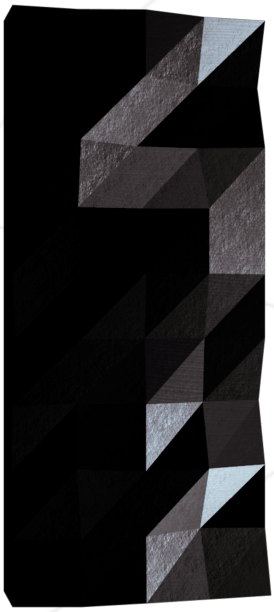

Piscina C

Ziua 11

Staff Academy+Plus contact@academyplus.ro

Sumar: Acest document este subiectul zilei 11 a piscinei C din cadrul Academy+Plus.

Cuprins

I	Instructiuni	2
II	Preambul	4
III	Exercitiu 00 : ft_create_elem	6
IV	Exercitiu 01 : ft_list_push_back	7
V	Exercitiu 02 : ft_list_push_front	8
VI	Exercitiu 03 : ft_list_size	9
VII	Exercitiu 04 : ft_list_last	10
VIII	Exercitiu 05 : ft_list_push_params	11
IX	Exercitiu 06 : ft_list_clear	12
X	Exercitiu 07 : ft_list_at	13
XI	Exercitiu 08 : ft_list_reverse	14
XII	Exercitiu 09 : ft_list_foreach	15
XIII	Exercitiu 10 : ft_list_foreach_if	16
XIV	Exercitiu 11 : ft_list_find	17
XV	Exercitiu 12 : ft_list_remove_if	18
XVI	Exercitiu 13 : ft_list_merge	19
XVII	Exercitiu 14 : ft_list_sort	20
XVIII	Exercitiu 15 : ft_list_reverse_fun	21
XIX	Exercitiu 16 : ft_sorted_list_insert	22
XX	Exercitiu 17 : ft_sorted_list_merge	23

Capitolul I

Instructiuni

- Utilizati doar aceaste pagini ca referinta; nu plecati urechea la zgomotul de pe coridor.
- Subiectul se poate schimba cu cel mult o ora inainte de incepere.
- Fiti atenti la drepturile pe care le aveti asupra fisierelor si directoarelor.
- Trebuie sa urmati procedurile de parcurgere pentru toate exercitiile voastre.
- Exercitiile voastre vor fi corectate de colegii vostri de piscina.
- Pe linga colegii vostri, veti fi corectati de un program numit Moulinette.
- Aplicatia Moulinette este foarte stricta in notare. Ea este total automatizata. Este imposibil sa comentati in legatura cu nota primita. Fiti foarte rigurosi pentru a evita surprizele.
- Moulinette nu e foarte desteapta. Ea nu poate intelege codul care nu respecta Standardele de scriere a codului (Norme).
- Utilizarea unei functii interzise este un caz de inselaciune (trisare). Toate aceste cazuri sunt sanctionate cu nota `-42`.
- Daca ft_putchar() este o functie valida, veti compila fisierul `ft_putchar.c`.
- Nu trebuie sa creati o functie main() decat atunci cand vi se cere sa scrieti un program.
- Exercitiile sunt strict ordonate de la cele simple spre cele complexe. In nici un caz nu vom lua in considerare un exercitiu complex rezolvat daca unul anterior, mai simplu, nu a fost rezolvat perfect.
- Aplicatia Moulinette se compileaza cu flag-urile: -Wall -Wextra -Werror.
- Daca programul vostru nu se compileaza, veti primi nota `0`.

- Nu lasati in directorul de lucru niciun fisier, altul decat cele specificate de enuntul exercitiului.
- Aveti intrebari? Intrebati-l pe vecinul din dreapta. Daca nu, incercati la cel din stanga.
- Manualele voastre de referinta sunt `Google / man / Internet / ...`.
- Puteti folosi forumul de pe Intranet pentru discutii legate de Piscina!
- Cititi cu atentie exemplele. Va pot oferi informatii suplimentare pentru elementele neclare din enunt...
- Reflectati la asta. Aveti mare grija!
- Pentru exercitiile pe liste, von utiliza structura urmatoare:

```
typedef struct          s_list
{
  struct s_list         *next;
  void                  *data;
}                       t_list;
```

- Va trebui sa puneti aceasta structura intr-un fisier `ft_list.h` si sa-l utilizati la fiecare exercitiu.
- Incepand cu exercitiul 01 vom utiliza ft_create_elem; luati masurile necesare (ar putea fi interesant sa fie inclus un prototip in ft_list.h...).

Capitolul II

Preambul

SPOILER ALERT
NU CITITI PAGINA URMATOARE

Ai cerut.

- In `Star Wars`, Dark Vador este parintele lui Luke Skywalker.
- In `The Usual Suspects`, Verbal este Keyser Soze.
- In `Fight Club`, Tyler Durden si naratorul aceeasi persoana.
- In `Sixième Sens`, Bruce Willis e mort de la inceput.
- In `Les Autres`, locatarii casei sunt sunt fantomele si vice-versa.
- In `Bambi`, mama lui Bambi moare.
- In `Le Village`, monstrii sunt satenii si actiunea se situeaza, in realitate, in epoca noastra.
- In `Harry Potter`, Dumbledore moare.
- In `La Planète des Singes`, actiunea se desfasoara pe Pamant.
- In `Le Trône de Fer`, Robb Stark si Joffrey Baratheon chiar din noaptea nuntii.
- In `Twilight`, vampirii vampires stralucesc in soare.
- In `Stargate SG-1, Saison 1, Episode 18`, O'Neill si Carter sunt in Antarctica.
- In `The Dark Knight Rises`, Miranda Tate este Talia Al'Gul.
- In `Super Mario Bros`, printesa este intr-un alt castel.

Capitolul III

Exercitiu 00 : ft_create_elem

	Exercitiu: 00
ft_create_elem	
Director de lucru: *ex*00/	
Fisier(e) de iesire: `ft_create_elem.c, ft_list.h`	
Functii autorizate: `malloc`	
Observatii: `n/a`	

- Scrieti functia `ft_create_elem` care creeaza un nou element de tip `t_list`.
- Ea trebuie sa asigneze campului `data` din structura, valoarea parametrului transmis si campului `*next` valoarea NULL.
- Ea trebuie sa aiba prototipul urmator:

```
t_list        *ft_create_elem(void *data);
```

Capitolul IV

Exercitiu 01 : ft_list_push_back

	Exercitiu: 01
ft_list_push_back	
Director de lucru: *ex*01/	
Fisier(e) de iesire: `ft_list_push_back.c, ft_list.h`	
Functii autorizate: `ft_create_elem`	
Observatii: `n/a`	

- Scrieti functia `ft_list_push_back` care adauga la sfarsitul listei un nou element de tip `t_list`.
- Ea trebuie sa asigneze campului `data` din structura `t_list` valoarea transmisa ca parametru.
- Ea va actualiza, daca e necesar, pointer-ul spre inceputul listei.
- Ea va trebui sa aiba prototipul urmator:

```
void        ft_list_push_back(t_list **begin_list, void *data);
```

Capitolul V

Exercitiu 02 : ft_list_push_front

	Exercitiu: 02
ft_list_push_front	
Director de lucru: *ex*02/	
Fisier(e) de iesire: `ft_list_push_front.c, ft_list.h`	
Functii autorizate: `ft_create_elem`	
Observatii: `n/a`	

- Scrieti functia `ft_list_push_front` care adauga la inceputul listei un element nou de tip `t_list`.
- Ea trebuie sa asigneze `data` parametrului furnizat.
- Ea va actualiza, daca e necesar, pointer-ul spre inceputul listei.
- Ea trebuie sa aiba prototipul urmator:

```
void        ft_list_push_front(t_list **begin_list, void *data);
```

Capitolul VI

Exercitiu 03 : ft_list_size

	Exercitiu: 03
ft_list_size	
Director de lucru: *ex*03/	
Fisier(e) de iesire: `ft_list_size.c, ft_list.h`	
Functii autorizate: `Niciuna`	
Observatii: `n/a`	

- Scrieti functia `ft_list_size` care returneaza numarul de elemente dintr-o lista.
- Ea va trebui sa aiba prototipul urmator:

```
int ft_list_size(t_list *begin_list);
```

Capitolul VII

Exercitiu 04 : ft_list_last

	Exercitiu: 04
ft_list_last	
Director de lucru: *ex*04/	
Fisier(e) de iesire: `ft_list_last.c, ft_list.h`	
Functii autorizate: `Niciuna`	
Observatii: `n/a`	

- Scrieti functia `ft_list_last` care returneaza ultimul element dintr-o lista.
- Ea trebuie sa aiba prototipul urmator:

```
t_list *ft_list_last(t_list *begin_list);
```

Capitolul VIII

Exercitiu 05 : ft_list_push_params

	Exercitiu: 05
ft_list_push_params	
Director de lucru: $ex05/$	
Fisier(e) de iesire: `ft_list_push_params.c, ft_list.h`	
Functii autorizate: `ft_create_elem`	
Observatii: `n/a`	

- Scrieti functia `ft_list_push_params` care creeaza o noua lista pe baza parametrilor transmisi in linia de comanda.
- Primul argument se va regasi la sfarsitul listei.
- Adresa primului element al listei este returnata.
- Ea trebuie sa aiba prototipul urmator:

```
t_list *ft_list_push_params(int ac, char **av);
```

Capitolul IX

Exercitiu 06 : ft_list_clear

	Exercitiu: 06
ft_list_clear	
Director de lucru: *ex*06/	
Fisier(e) de iesire: `ft_list_clear.c, ft_list.h`	
Functii autorizate: `free`	
Observatii: `n/a`	

- Scrieti functia `ft_list_clear` care sterge elementele listei.
- Ea asigneaza apoi pointer-ul spre lista va primi in final valoarea NULL.
- Ea trebuie sa aiba prototipul urmator:

```
void ft_list_clear(t_list **begin_list);
```

Capitolul X

Exercitiu 07 : ft_list_at

	Exercitiu: 07
ft_list_at	
Director de lucru: *ex*07/	
Fisier(e) de iesire: `ft_list_at.c, ft_list.h`	
Functii autorizate: `Niciuna`	
Observatii: `n/a`	

- Scrieti functia `ft_list_at` care returneaza elementul n al listei.
- Ea returneaza un pointer nul in caz de eroare.
- Functia va avea prototipul urmator:

```
t_list *ft_list_at(t_list *begin_list, unsigned int nbr);
```

Capitolul XI

Exercitiu 08 : ft_list_reverse

	Exercitiu: 08
ft_list_reverse	
Director de lucru: $ex08/$	
Fisier(e) de iesire: `ft_list_reverse.c, ft_list.h`	
Functii autorizate: `Niciuna`	
Observatii: `n/a`	

- Scrieti functia `ft_list_reverse` care inverseaza ordinea elementelor unei liste. Se admite doar lucrul cu pointer-ii.
- Ea trebuie sa aiba prototipul urmator:

```
void ft_list_reverse(t_list **begin_list);
```

Capitolul XII

Exercitiu 09 : ft_list_foreach

	Exercitiu: 09
ft_list_foreach	
Director de lucru: *ex*09/	
Fisier(e) de iesire: `ft_list_foreach.c, ft_list.h`	
Functii autorizate: `Niciuna`	
Observatii: `n/a`	

- Scrieti functia `ft_list_foreach` care aplica o functie data ca parametru la informatia continuta in fiecare element din lista.
- Ea va trebui sa aiba prototipul urmator:

```
void ft_list_foreach(t_list *begin_list, void (*f)(void *));
```

- Functia referentiata de `f` va fi utilizata in modul urmator:

```
(*f)(list_ptr->data);
```

Capitolul XIII

Exercitiu 10 : ft_list_foreach_if

	Exercitiu: 10
ft_list_foreach_if	
Director de lucru: $ex10/$	
Fisier(e) de iesire: `ft_list_foreach_if.c, ft_list.h`	
Functii autorizate: `Niciuna`	
Observatii: `n/a`	

- Scrieti functia `ft_list_foreach_if` care aplica o functie data ca parametru asupra informatiei continute in anumitenoduri din lista. Folosind o valoare de referinta *data_ref si o functie de comparare, vom selecta doar acele noduri care sunt "egale" cu informatia de referinta.

- Ea trebuie sa aiba prototipul urmator:

```
void        ft_list_foreach_if(t_list *begin_list, void (*f)(void *), void
*data_ref, int (*cmp)(void *, void *))
```

- Functiile referentiate de `f` si `cmp` vor fi utilizate in modul urmator:

```
(*f)(list_ptr->data);
(*cmp)(list_ptr->data, data_ref);
```

`Functia cmp ar putea fi de exemplu ft_strcmp...`

Capitolul XIV

Exercitiu 11 : ft_list_find

	Exercitiu: 11
ft_list_find	
Director de lucru: $ex11/$	
Fisier(e) de iesire: `ft_list_find.c, ft_list.h`	
Functii autorizate: `Niciuna`	
Observatii: `n/a`	

- Scrieti functia `ft_list_find` care returneaza adresa primului element a carui valoare este egala cu valoarea de referinta.
- Ea trebuie sa aiba prototipul urmator:

```
t_list *ft_list_find(t_list *begin_list, void *data_ref, int (*cmp)());
```

Capitolul XV

Exercitiu 12 : ft_list_remove_if

	Exercitiu: 12
ft_list_remove_if	
Director de lucru: $ex12/$	
Fisier(e) de iesire: `ft_list_remove_if.c, ft_list.h`	
Functii autorizate: `free`	
Observatii: `n/a`	

- Scrieti functia `ft_list_remove_if` care sterge din lista toate elementele a caror valoare este "egala" cu valoarea de referinta.

- Ea trebuie sa aiba prototipul urmator:

```
void ft_list_remove_if(t_list **begin_list, void *data_ref, int (*cmp)());
```

Capitolul XVI

Exercitiu 13 : ft_list_merge

	Exercitiu: 13
ft_list_merge	
Director de lucru: $ex13/$	
Fisier(e) de iesire: `ft_list_merge.c, ft_list.h`	
Functii autorizate: `Niciuna`	
Observatii: `n/a`	

- Scrieti functia `ft_list_merge` care pune elementele unei liste `begin2` la sfarsitul unei alte liste `begin1`.
- Crearea de noi elemente de tip t_list nu este permisa.
- Ea trebuie sa aiba prototipul urmator:

```
void ft_list_merge(t_list **begin_list1, t_list *begin_list2);
```

Capitolul XVII

Exercitiu 14 : ft_list_sort

	Exercitiu: 14
ft_list_sort	
Director de lucru: *ex*14/	
Fisier(e) de iesire: `ft_list_sort.c, ft_list.h`	
Functii autorizate: `Niciuna`	
Observatii: `n/a`	

- Scrieti functia `ft_list_sort` care ordoneaza crescator continutul unei liste, prin compararea a doua elemente prin intermediul unei functii de comparare a valorii a doua elemente.

- Ea trebuie sa aiba prototipul urmator:

```
void ft_list_sort(t_list **begin_list, int (*cmp)());
```

`Functia cmp ar putea fi de exemplu ft_strcmp.`

Capitolul XVIII

Exercitiu 15 : ft_list_reverse_fun

	Exercitiu: 15
ft_list_reverse_fun	
Director de lucru: $ex15/$	
Fisier(e) de iesire: `ft_list_reverse_fun.c, ft_list.h`	
Functii autorizate: `Niciuna`	
Observatii: `n/a`	

- Scrieti functia `ft_list_reverse_fun` care inverseaza ordinea elementelor unei liste. Se vor pute utiliza doar pointer-i.

- Ea trebuie sa aiba prototipul urmator:

```
void ft_list_reverse_fun(t_list *begin_list);
```

Capitolul XIX

Exercitiu 16 : ft_sorted_list_insert

	Exercitiu: 16
ft_sorted_list_insert	
Director de lucru: *ex*16/	
Fisier(e) de iesire: `ft_sorted_list_insert.c, ft_list.h`	
Functii autorizate: `ft_create_elem`	
Observatii: `n/a`	

- Scrieti functia `ft_sorted_list_insert` care creeaza un nou element si il insereaza intr-o lista ordonata astfel incat lista sa ramana ordonata crescator.
- Ea trebuie sa aiba prototipul urmator:

```
void ft_sorted_list_insert(t_list **begin_list, void *data, int (*cmp)());
```

Capitolul XX

Exercitiu 17 : ft_sorted_list_merge

	Exercitiu: 17
ft_sorted_list_merge	
Director de lucru: *ex*17/	
Fisier(e) de iesire: `ft_sorted_list_merge.c, ft_list.h`	
Functii autorizate: `Niciuna`	
Observatii: `n/a`	

- Scrieti functia `ft_sorted_list_merge` care integreaza elementele unei liste ordonate `begin2` intr-o alta lista ordonata `begin1`, astfel incat lista `begin1` ramane ordonata crescator.

- Ea trebuie sa aiba prototipul urmator:

```
void ft_sorted_list_merge(t_list **begin_list1, t_list *begin_list2, int (*cmp)());
```