**МИНОБРНАУКИ РОССИИ**
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**
**Кафедра** МО ЭВМ

# ОТЧЕТ

**по курсовой работе**

**по дисциплине «**Программирование»

**Тема:** Генерация отчетов

гр. 7382 _____ Глазунов С.А.

Преподаватель _____ Кринкин К.В.

Санкт-Петербург

2018

**ЗАДАНИЕ**

**НА КУРСОВУЮ РАБОТУ**

Студент Глазунов С.А.

Группа 7382

Тема работы : Генерация отчетов

Исходные данные: В качестве основы для курсовой работы используется код лабораторной работы No4.

Содержание пояснительной записки:«Введение», «Заключение», «Список использованных источников»

Предполагаемый объем пояснительной записки:

Не менее 5 страниц.

Дата выдачи задания: 28.11.2017

Дата сдачи реферата: 23.12.2017

Дата защиты реферата: 23.12.2017

Студент _____ Глазунов С.А.

Преподаватель _____ Кринкин К.В.

**Аннотация.**

Необходимо, имея код лабораторной работы No4, реализовать алгоритм,который делает следующие преобразования со списком:

• Отсортировать список по невозрастанию по полю year в этом списке;

• Менять местами элементы не затрагивая поля, кроме тех, что указывают наследующие и предыдущие элементы;

Пишется две функции, которые производят все эти преобразования и возвращают головной элемент списка.

Пишется две функции, которые производят все эти преобразования и возвращают головной элемент списка.

**Введение.**

Необходимо, имея код лабораторной работы No4, реализовать алгоритм,который делает следующие преобразования со списком:

• Отсортировать список по невозрастанию по полю year в этом списке;

• Менять элементы не трогая поля, кроме тех, которые указывают на следующий и на предыдущий элемент

**Реализация функций.**

На вход функции, которая названа llist_bubble_sort () подается адрес первого элемента списка, который условно назван “head”. Внутри llist_bubble_sort () была использована функция swap(), на вход которой подается 2 элемента(подразумевается, что первый аргумент является левым элементом списка второго аргумента) и адрес “head”, а также функция count(), написанная ранее для лабораторной работы No4, чтобы подсчитать количество элементов списка.

**Сортировка списка.**

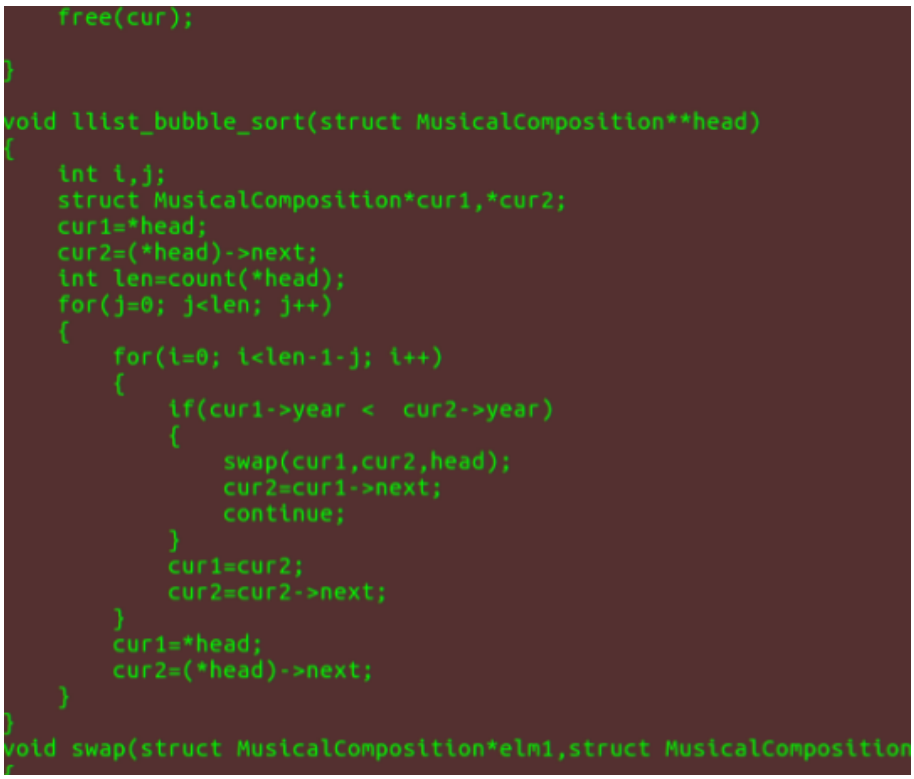
Изначально нужно определить, сколько элементов будет в списке. Для этого вызовем функцию count(). Также потребуется 2 новый указателя , которые изначально будут указывать на “head” и “head->next” соответственно.

**Реализация.**

Создаем переменную len, которая будет хранить количество элементов списка, и присваиваем ей значение:

Далее инициализируем другие переменные, которые будут являться указателями на “head” и “head->next”: Struct MusicalComposition* cur1*cur2, далее начинается первый цикл:

Переменная j<len-1 , потому что указатель cur2 из пункта указывает наследующий элемент первого. Когда бы наступила последняя итерация цикла то cur2 указывал уже на несуществующий элемент списка, что могло бы повлечь за собой ошибку сегментации. Дальше во внутри 1-ого цикла есть еще и второй цикл, который при каждой итерации первого цикла проходит по len-1-j элементов списка. Уже во 2-ом цикле происходит сравнение поля year двух элементов списка. Если cur1->year меньше cur2->year , то вызывается функция swap, которая меняет элементы местами (См. Рис. 1).



```
    free(cur);

}

void llist_bubble_sort(struct MusicalComposition**head)
{
    int i,j;
    struct MusicalComposition*cur1,*cur2;
    cur1=*head;
    cur2=(*head)->next;
    int len=count(*head);
    for(j=0; j<len; j++)
    {
        for(i=0; i<len-1-j; i++)
        {
            if(cur1->year <  cur2->year)
            {
                swap(cur1,cur2,head);
                cur2=cur1->next;
                continue;
            }
            cur1=cur2;
            cur2=cur2->next;
        }
        cur1=*head;
        cur2=(*head)->next;
    }
}
void swap(struct MusicalComposition*elm1,struct MusicalComposition
```

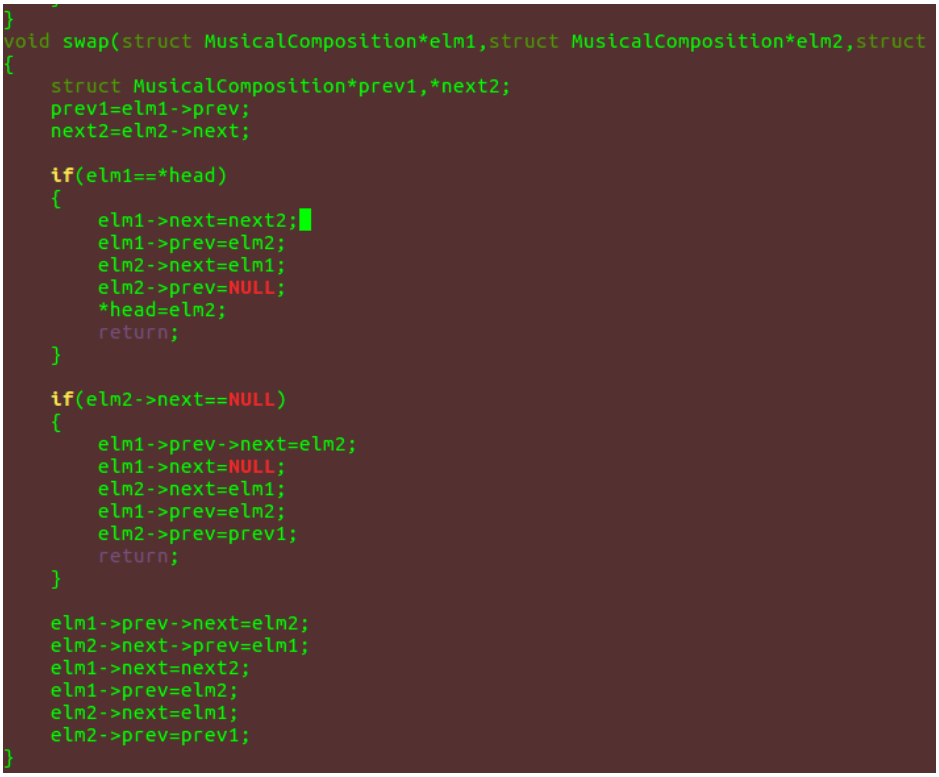
Рисунок 1.

Теперь рассмотрим функцию swap. В функции swap надо рассмотреть 3 случая:

• Меняется “head” со следующим элементом;

• Меняется два элемента списка, ни один из которых не является головой или хвостом списка;

• Меняется хвост и предыдущий элемент. Для каждого случая нужен свой блок кода, состоящий из двух условий, поэтому в каждом блоке есть команда return – это сделано для того, чтобы Функция не проверяла лишние условия, когда заведомо следующие условия ложны (См. Рис. 2).



```
}
void swap(struct MusicalComposition*elm1,struct MusicalComposition*elm2,struct
{
    struct MusicalComposition*prev1,*next2;
    prev1=elm1->prev;
    next2=elm2->next;

    if(elm1==*head)
    {
        elm1->next=next2;
        elm1->prev=elm2;
        elm2->next=elm1;
        elm2->prev=NULL;
        *head=elm2;
        return;
    }

    if(elm2->next==NULL)
    {
        elm1->prev->next=elm2;
        elm1->next=NULL;
        elm2->next=elm1;
        elm1->prev=elm2;
        elm2->prev=prev1;
        return;
    }

    elm1->prev->next=elm2;
    elm2->next->prev=elm1;
    elm1->next=next2;
    elm1->prev=elm2;
    elm2->next=elm1;
    elm2->prev=prev1;
}
```

Рисунок 2.

**Принцип работы функции swap:**

Так как мы работаем в линейном двусвязном списке, то для того, чтобы поменять два элемента местами в общем случае надо поменять 6 связей в сумме. По две связи у элементов, которые должны поменяться местами и по одной связи у элементов, которые до cur1 и после cur2. В cur1->prev надо поменять указатель next с cur1 на cur2. В cur2->next надо поменять указатель prev с cur2 на cur1. В cur1 надо поменять prev с cur1->prev на cur2, next поменять с cur2 на cur2->next(аналогично cur1->next->next). В cur2 надо поменять prev с cur1 на cur1->prev и next поменять с cur2->next на cur1. Следует отметить, что менять адреса самих указателей не рекомендуется, потому что возможна потеря адреса какого элемента и в последствии приведет к неправильной работе программы. случаях когда меняется голова или хвост следует лишь отметить, что один из указателей будет указывать на NULL, и поэтому код программы для этих случаев будет лишь слегка отличаться от общего случая.

**Как запускается программа.**

Вся программа состоит из нескольких файлов. Один файл-main. c, который и содержит код для выполнения основной задачи. Главные функции этого файла уже были описаны ранее в 4 лабораторной и в пункте 1 В файле “course. sh” содержится скрипт на языке bash. Благодаря этому скрипту можно передать основной программе (main. c) любой файл, который содержит текст и лежит в одной директории. Также в этом файле реализована утилита beep(См. Рис. 3). Она нужна для того, чтобы компьютер подал звуковой сигнал, когда программа завершится. Это сделано для удобства программиста, так как если список будет состоять из тысячи и больше элементов, то сортировка может занять некоторое время.

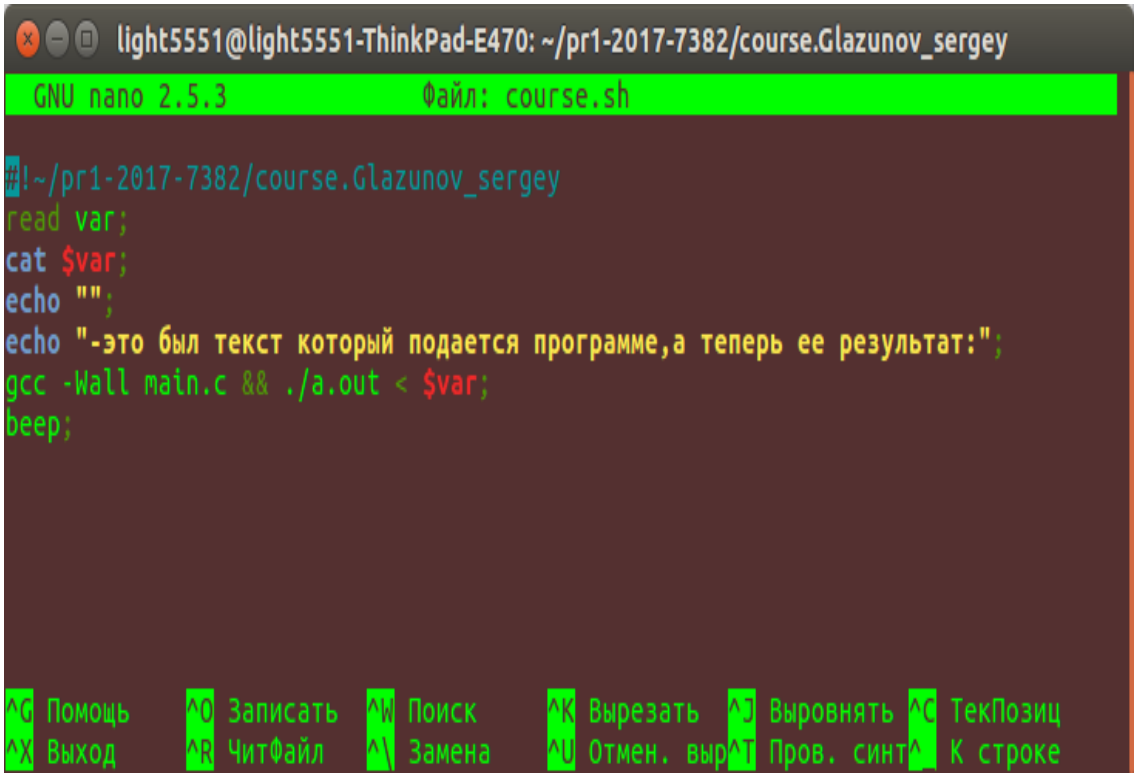



Рисунок 3.

**Демонстрация работы программы.**

Для того, чтобы показать работу функции llist_bubble_sort () используем код функции main из лабораторной работы No4 и исходные данные из нее же. Внутри функции main() происходит считывание значений и создание списка, состоящего из структур MusicalComposition, которые содержат: название группы, альбома, год выхода альбома и указатели на предыдущий и следующий элементы списка. Сначала проверяется возможен ли push в списке и подсчет элементов(См. Рис. 4).


Рисунок 4.

Дальше идет сортировка списка функцией llist_bubble_sort () и после удаления одного элемента(См. Рис. 5).

Рисунок 5.

**Заключение.**

Была поставлена задача – отсортировать список по убыванию поля year. Написанные функции llist_bubble_sort и swap успешно справляются с поставленной задачей

Была поставлена задача – отсортировать список по убыванию поля year. Написанные функции llist_bubble_sort и swap успешно справляются с поставленной задачей

# Приложение

## test.c

```
1 f
2 r
3 o
4 m
5
6 r
7 a
8 n
9 d
10 o
11 m
12
13 i
14 m
15 p
16 o
17 r
18 t
19
20 r
21 a
22 n
23 d
24 i
25 n
26 t
27
28
29
30 d
31 e
32 f
33
34 b
35 u
36 b
37 b
38 l
39 e
40 (
41 a
42 r
43 r
44 a
45 y
46 )
47 :
48
49
50
51
52
53 f
54 o
55 r
56
57 i
```

```
58
59 i
60 n
61
62 r
63 a
64 n
65 g
66 e
67 (
68 N
69 -
70 1
71 )
72 :
73
74
75
76
77
78
79
80
81
82 f
83 o
84 r
85
86 j
87
88 i
89 n
90
91 r
92 a
93 n
94 g
95 e
96 (
97 N
98 -
99 i
100 -
101 1
102 )
103 :
104
105
106
107
108
109
110
111
112
113
114
115
116
117 i
118 f
```

119
120 a
121 r
122 r
123 a
124 y
125 [
126 j
127 ]
128
129 >
130
131 a
132 r
133 r
134 a
135 y
136 [
137 j
138 +
139 1
140 ]
141 :
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159 b
160 u
161 f
162 f
163
164 =
165
166 a
167 r
168 r
169 a
170 y
171 [
172 j
173 ]
174
175
176
177
178
179

180
181
182
183
184
185
186
187
188
189
190
191 a
192 r
193 r
194 a
195 y
196 [
197 j
198 ]
199
200 =
201
202 a
203 r
204 r
205 a
206 y
207 [
208 j
209 +
210 1
211 ]
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229 a
230 r
231 r
232 a
233 y
234 [
235 j
236 +
237 1
238 ]
239
240 =

241
242 b
243 u
244 f
245 f
246
247
248
249 N
250
251 =
252
253 1
254 0
255
256 a
257
258 =
259
260 [
261 ]
262
263 f
264 o
265 r
266
267 i
268
269 i
270 n
271
272 r
273 a
274 n
275 g
276 e
277 (
278 N
279 )
280 :
281
282
283
284
285
286 a
287 .
288 a
289 p
290 p
291 e
292 n
293 d
294 (
295 r
296 a
297 n
298 d
299 i
300 n
301 t

```
302 (
303 1
304 ,
305
306 9
307 9
308 )
309 )
310
311
312
313 p
314 r
315 i
316 n
317 t
318 (
319 a
320 )
321
322 b
323 u
324 b
325 b
326 l
327 e
328 (
329 a
330 )
331
332 p
333 r
334 i
335 n
336 t
337 (
338 a
339 )
340
```

## test2.c

```
1 /
2 /
3 С
4 л
5 е
6 д
7 у
8 ю
9 щ
10 а
11 я
12
13 п
14 р
15 о
16 г
17 р
18 а
19 м
20 м
```

21 a
22
23 c
24 o
25 p
26 т
27 и
28 p
29 y
30 e
31 т
32
33 c
34 п
35 и
36 c
37 o
38 к
39
40 ч
41 и
42 c
43 e
44 л
45
46 и
47
48 в
49 ы
50 в
51 o
52 д
53 и
54 т
55
56 p
57 e
58 з
59 y
60 л
61 ь
62 т
63 a
64 т
65 :
66
67 #
68 i
69 n
70 c
71 l
72 u
73 d
74 e
75
76 <
77 s
78 t
79 d
80 i
81 o

```
82 .
83 h
84 >
85
86 #
87 i
88 n
89 c
90 l
91 u
92 d
93 e
94
95 <
96 s
97 t
98 d
99 l
100 i
101 b
102 .
103 h
104 >
105
106
107 i
108 n
109 t
110
111 n
112 u
113 m
114 [
115 1
116 0
117 ]
118
119 =
120
121 {
122
123 1
124 ,
125 3
126 ,
127 6
128 ,
129 5
130 ,
131 8
132 ,
133 7
134 ,
135 9
136 ,
137 6
138 ,
139 2
140 ,
141 0
142
```

```
143 }
144 ;
145
146
147 i
148 n
149 t
150
151 c
152 o
153 m
154 p
155
156 (
157 c
158 o
159 n
160 s
161 t
162
163 i
164 n
165 t
166
167 *
168 ,
169
170 c
171 o
172 n
173 s
174 t
175
176 i
177 n
178 t
179
180 *
181 )
182 ;
183
184
185 i
186 n
187 t
188
189 m
190 a
191 i
192 n
193 (
194 v
195 o
196 i
197 d
198 )
199
200 {
201
202 i
203 n
```

```
204 t
205
206 i
207 ;
208
209 p
210 r
211 i
212 n
213 t
214 f
215 (
216 "
217 O
218 r
219 i
220 g
221 i
222 n
223 a
224 l
225
226 a
227 r
228 r
229 a
230 y
231 :
232
233 "
234 )
235 ;
236
237
238
239
240
241 f
242 o
243 r
244
245 (
246 i
247 =
248 0
249 ;
250
251 i
252 <
253 1
254 0
255 ;
256
257 i
258
259 +
260
261 +
262 )
263
264
```

```
265
266
267
268
269 {
270
271
272
273
274
275
276
277
278
279 p
280 r
281 i
282 n
283 t
284 f
285 (
286 "
287 %
288 d
289
290 "
291 ,
292 n
293 u
294 m
295 [
296 i
297 ]
298 )
299 ;
300
301
302
303
304
305 }
306
307
308 p
309 r
310 i
311 n
312 t
313 f
314
315 (
316 "
317 \
318 n
319 "
320 )
321 ;
322
323
324 q
325 s
```

```
326 o
327 r
328 t
329 (
330 n
331 u
332 m
333 ,
334
335 1
336 0
337 ,
338
339 s
340 i
341 z
342 e
343 o
344 f
345
346 (
347 i
348 n
349 t
350 )
351 ,
352
353 (
354 i
355 n
356 t
357 (
358 *
359 )
360
361 (
362 c
363 o
364 n
365 s
366 t
367
368 v
369 o
370 i
371 d
372
373 *
374 ,
375
376 c
377 o
378 n
379 s
380 t
381
382 v
383 o
384 i
385 d
386
```

```
387 *
388 )
389 )
390
391 c
392 o
393 m
394 p
395 )
396 ;
397
398
399 p
400 r
401 i
402 n
403 t
404 f
405 (
406 "
407 S
408 o
409 r
410 t
411 e
412 d
413
414 a
415 r
416 r
417 a
418 y
419 :
420
421 "
422 )
423 ;
424
425
426
427
428
429 f
430 o
431 r
432 (
433 i
434
435 =
436
437 0
438 ;
439
440 i
441
442 <
443 1
444 0
445 ;
446
447 i
```

```
448
449 +
450
451 +
452
453 )
454
455
456
457
458
459
460
461
462
463
464 p
465 r
466 i
467 n
468 t
469 f
470 (
471 "
472 %
473 d
474
475 "
476 ,
477
478 n
479 u
480 m
481 [
482 i
483 ]
484 )
485 ;
486
487 r
488 e
489 t
490 u
491 r
492 n
493
494 0
495 ;
496
497 }
498
499
500 /
501 *
502
503 c
504 p
505 a
506 в
507 н
508 e
```

509 н
510 и
511 е
512
513 д
514 в
515 у
516 х
517
518 ц
519 е
520 л
521 ы
522 х
523
524 *
525 /
526
527 i
528 n
529 t
530
531 c
532 o
533 m
534 p
535
536 (
537 c
538 o
539 n
540 s
541 t
542
543 i
544 n
545 t
546
547 *
548 i
549 ,
550
551 c
552 o
553 n
554 s
555 t
556
557 i
558 n
559 t
560
561 *
562 j
563 )
564
565 {
566
567
568
569 r

```
570 e
571 t
572 u
573 r
574 n
575
576 *
577 i
578
579 -
580
581 *
582 j
583 ;
584
585 }
586
```