

# Отчет по лабораторной работе № 11 по курсу “Фундаментальная информатика”

Студент группы М80-109Б-22 Серякова Александра Андреевна, № 17

Работа выполнена: «02» октября 2022г.

Преподаватель: каф. 806 Сысоев Максим Алексеевич

Отчет сдан « » \_\_\_\_\_ 20\_\_ г., итоговая оценка \_\_\_\_\_

Подпись преподавателя \_\_\_\_\_

1. **Тема:** Алгоритмы и структуры данных
2. **Цель работы:** Составить программу на языке Си, выполняющую анализ и обработку вводимого текста в соответствии с выданным преподавателем вариантом задания.
3. **Задание (вариант № 29):** 29. Удалить все десятичные числа, не превышающие INT\_MAX
4. **Оборудование (студента):**

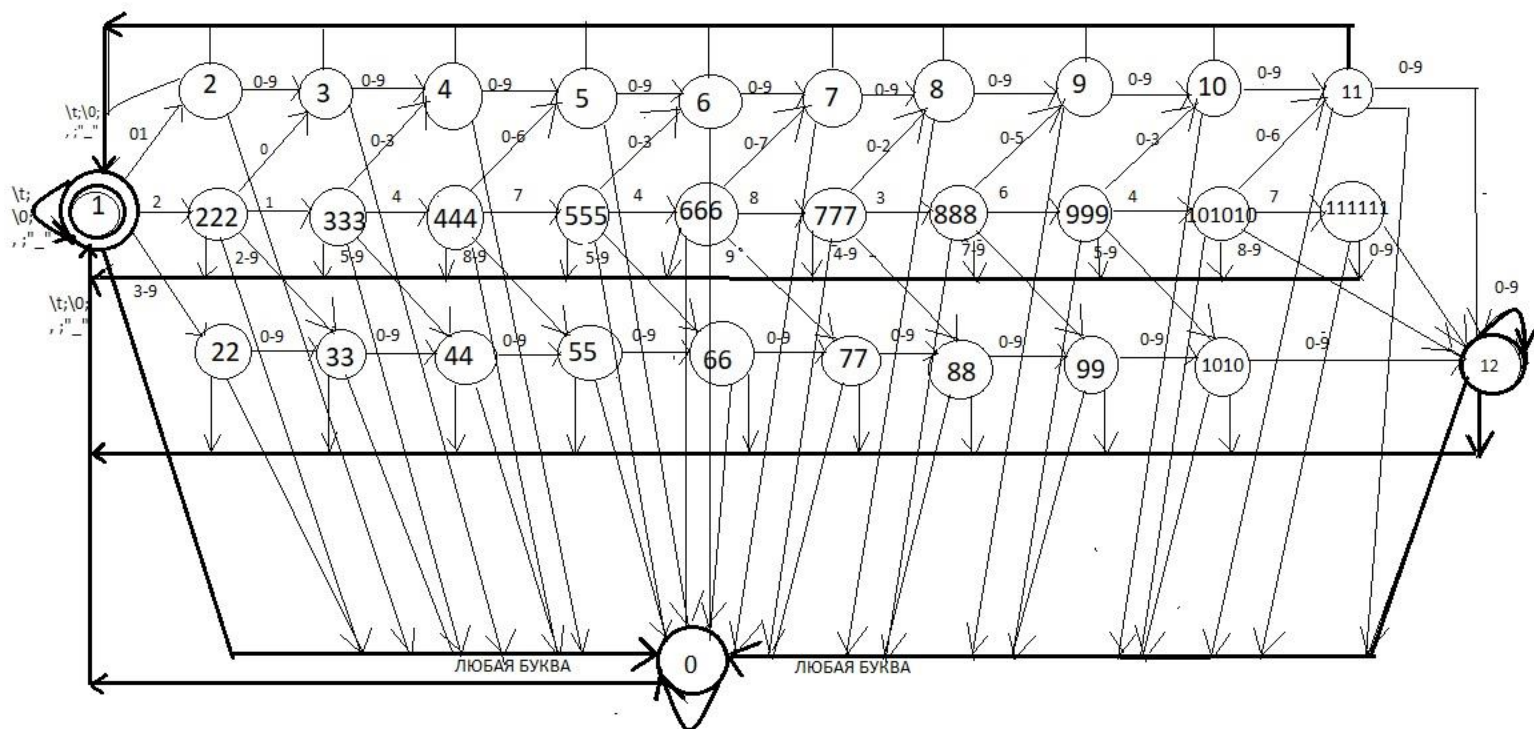
*Процессор AMD Ryzen 5 5500U with Radeon Graphics @ 2.100GHz с ОП 9812 Мб, SSD 512 Гб.  
Монитор 1920x1080*

5. **Программное обеспечение (студента):**

Операционная система семейства: linux, наименование: Arch x86\_64  
интерпретатор команд: bash версия 5.1.16  
Система программирования -- версия --, редактор текстов neo vim версия 0.7.2  
Утилиты операционной системы mkdir, cd, touch, ls, echo, cat, find, grep, rm, chmod, bash, pwd  
Прикладные системы и программы –  
Местонахождение и имена файлов программ и данных на домашнем компьютере  
/home/taida/Programming/MAI\_labs/lab5

6. **Идея, метод, алгоритм** решения задачи (в формах: словесной, псевдокода, графической [блок-схема, диаграмма, рисунок, таблица] или формальные спецификации с пред- и постусловиями)

INT MAX = 2147483647  
 № элемента 0 1 2 3 4 5 6 7 8 9



**7. Сценарий выполнения работы** [план работы, первоначальный текст программы в черновике (можно на отдельном листе) и тесты либо соображения по тестированию].

Проводилось unit-тестирование, исходный код тестов приложен в пункте №8.

Тесты:

описание	Входные данные	результат
Когда слова-это числа через пробел	Input: 4567887656789 4567656 567 4 4567765678767	4567887656789 4567765678767
Когда слова-это буквы и через пробел	Input: rtyuiutghjkgghjk rtyu f fghjk	rtyuiutghjkgghjk rtyu f fghjk
Когда и слова, и числа через пробел	Input: r456 5 567 45678765677 56776ty	r456 45678765677 56776ty
Когда и слова, и числа через запятую и пробел	Input: 456765fg,45676567656776567,677 567898 fgh56787 , 45678567876545677	456765fg 45676567656776567 fgh56787 45678567876545677

**8. Распечатка протокола** (подклеить листинг окончательного варианта программы с тестовыми примерами, подписанный преподавателем).

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <limits.h>
4  #include <string.h>
5
6  int all_number(char c)
7  {
8      return (c>= '0' && c<='9');
9  }
10
11 int less_number(char c, char num_arr)
12 {
13     return (c>= '0' && c<='9' && c< num_arr);
14 }
15
16 int more_number(char c, char num_arr)
17 {
18     return (c>= '0' && c<='9' && c> num_arr);
19 }
20
21 int equal_number(char c, char num_arr)
22 {
23     return (c>= '0' && c<='9' && c== num_arr);
24 }
25
26 int gap(char c)
27 {
28     return (c== ' ' || c== '\t' || c== ',' || c== '\n');
29 }
30
31 int bukv(char c)
32 {
33     return (!(all_number(c)) && !(gap(c)));
34 }
35
36 int main(void)
37 {
38     printf("%d\n", INT_MAX);
39     int int_max = INT_MAX;
40
41     int i= 0;
42     int tarr[10] = {0};
43     int arr[10] = {};
44
45     while (int_max != 0)
46     {
47         tarr[i] = int_max % 10; // MAXIMUM НАХОДИМ НА КАЖДОМ ШАГЕ
48         ++i;
49         int_max /= 10;
50     }
51     for (; i!= 0; --i)
52     {
53         arr[10-i] = tarr[i-1]; // ЗАПИСАЛИ НАШ МАКС В АРР
54     }
55     printf("Input: ");
56
57     int state = 1;
58     int k;
59     char symbol;
60     char all[20] = " "; //наш буфер хранения для ввода
61     int top = 0; //top - это позиция хранения all
62     while ((symbol = getchar()) != EOF) {
63         switch (state) {
64             case 1:
65                 if (less_number(symbol, arr[0])) {
66                     top = 0;
67                     memset (all, ' ', 20-1);
68                     all[top] = symbol;
69                     ++top;
70                     state = 2;
71                 } else if (equal_number(symbol, arr[0])) {
72                     top = 0;
73                     memset (all, ' ', 20-1);
74                     all[top] = symbol;
75                     ++top;
76                     state = 222;
77                 } else if (more_number(symbol, arr[0])) {
78                     top = 0;
79                     memset (all, ' ', 20-1);
80                     all[top] = symbol;
81                     ++top;
82                     state = 22;
83                 } else if (gap(symbol)) {
84                     state = 1;
85                 } else if (bukv(symbol)) {
86                     top = 0;
87                     memset (all, ' ', 20-1);
88                     all[top] = symbol;
89                     ++top;
90                     state = 0;
91                 }
92             break;
93             case 0:
94                 if (gap(symbol)) {
95                     printf("%s\n", all);
96                     state= 1;
97                 }
98             }
99     }
```

```

100         all[top] = symbol;
101         ++top;
102         state = 0;
103     }
104     break;
105 case 2:
106     if (all_number(symbol)){
107         all[top] = symbol;
108         ++top;
109         state = 3;
110     }else if (gap(symbol)){
111         state = 1;
112     }else if (bukv(symbol)){
113         all[top] = symbol;
114         ++top;
115         state = 0;
116     }
117     break;
118 case 222:
119     if (less_number(symbol, arr[1])){
120         all[top] = symbol;
121         ++top;
122         state = 3;
123     }else if (equal_number(symbol, arr[1])){
124         all[top] = symbol;

```

```

125         ++top;
126         state = 333;
127     }else if (more_number(symbol, arr[1])){
128         all[top] = symbol;
129         ++top;
130         state = 33;
131     }else if (gap(symbol)){
132         state = 1;
133     }else if (bukv(symbol)){
134         all[top] = symbol;
135         ++top;
136         state = 0;
137     }
138     break;
139 case 22:
140     if (all_number(symbol)){
141         all[top] = symbol;
142         ++top;
143         state = 33;
144     }else if (gap(symbol)){
145         state = 1;
146     }else if (bukv(symbol)){
147         all[top] = symbol;
148         ++top;
149         state = 0;

```

```

150     }
151     break;
152
153 case 3:
154     if (all_number(symbol)){
155         all[top] = symbol;
156         ++top;
157         state = 4;
158     }else if (gap(symbol)){
159         state = 1;
160     }else if (bukv(symbol)){
161         all[top] = symbol;
162         ++top;
163         state = 0;
164     }
165     break;
166 case 333:
167     if (less_number(symbol, arr[2])){
168         all[top] = symbol;
169         ++top;
170         state = 4;
171     }else if (equal_number(symbol, arr[2])){
172         all[top] = symbol;
173         ++top;
174         state = 444;

```

```

175     }else if (more_number(symbol, arr[2])){
176         all[top] = symbol;
177         ++top;
178         state = 44;
179     }else if (gap(symbol)){
180         state = 1;
181     }else if (bukv(symbol)){
182         all[top] = symbol;
183         ++top;
184         state = 0;
185     }
186     break;
187 case 33:
188     if (all_number(symbol)){
189         all[top] = symbol;
190         ++top;
191         state = 44;
192     }else if (gap(symbol)){
193         state = 1;
194     }else if (bukv(symbol)){
195         all[top] = symbol;
196         ++top;
197         state = 0;
198     }
199     break;

```

```

200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224

```

```

case 4:
    if (all_number(symbol)){
        all[top] = symbol;
        ++top;
        state = 5;
    }else if (gap(symbol)){
        state = 1;
    }else if (bukv(symbol)){
        all[top] = symbol;
        ++top;
        state = 0;
    }
    break;
case 44:
    if (less_number(symbol,arr[3])){
        all[top] = symbol;
        ++top;
        state = 5;
    }else if (equal_number(symbol, arr[3])){
        all[top] = symbol;
        ++top;
        state = 55;
    }else if (more_number(symbol, arr[3])){
        all[top] = symbol;

```

```

225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249

```

```

        ++top;
        state = 55;
    }else if (gap(symbol)){
        state = 1;
    }else if (bukv(symbol)){
        all[top] = symbol;
        ++top;
        state = 0;
    }
    break;
case 44:
    if (all_number(symbol)){
        all[top] = symbol;
        ++top;
        state = 55;
    }else if (gap(symbol)){
        state = 1;
    }else if (bukv(symbol)){
        all[top] = symbol;
        ++top;
        state = 0;
    }
    break;

```

```

250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299

```

```

case 5:
    if (all_number(symbol)){
        all[top] = symbol;
        ++top;
        state = 6;
    }else if (gap(symbol)){
        state = 1;
    }else if (bukv(symbol)){
        all[top] = symbol;
        ++top;
        state = 0;
    }
    break;
case 55:
    if (less_number(symbol,arr[4])){
        all[top] = symbol;
        ++top;
        state = 6;
    }else if (equal_number(symbol, arr[4])){
        all[top] = symbol;
        ++top;
        state = 66;
    }else if (more_number(symbol, arr[4])){
        all[top] = symbol;
        ++top;
        state = 66;
    }else if (gap(symbol)){
        state = 1;
    }else if (bukv(symbol)){
        all[top] = symbol;
        ++top;
        state = 0;
    }
    break;
case 55:
    if (all_number(symbol)){
        all[top] = symbol;
        ++top;
        state = 66;
    }else if (gap(symbol)){
        state = 1;
    }else if (bukv(symbol)){
        all[top] = symbol;
        ++top;
        state = 0;
    }
    break;
case 6:
    if (all_number(symbol)){
        all[top] = symbol;

```

```

300         ++top;
301         state = 7;
302     }else if (gap(symbol)){
303         state = 1;
304     }else if (bukv(symbol)){
305         all[top] = symbol;
306         ++top;
307         state = 0;
308     }
309     break;
310 case 666:
311     if (less_number(symbol,arr[5])){
312         all[top] = symbol;
313         ++top;
314         state = 7;
315     }else if (equal_number(symbol, arr[5])){
316         all[top] = symbol;
317         ++top;
318         state = 777;
319     }else if (more_number(symbol, arr[5])){
320         all[top] = symbol;
321         ++top;
322         state = 77;
323     }else if (gap(symbol)){
324         state = 1;
325     }else if (bukv(symbol)){
326         all[top] = symbol;
327         ++top;
328         state = 0;
329     }
330     break;
331 case 66:
332     if (all_number(symbol)){
333         all[top] = symbol;
334         ++top;
335         state = 77;
336     }else if (gap(symbol)){
337         state = 1;
338     }else if (bukv(symbol)){
339         all[top] = symbol;
340         ++top;
341         state = 0;
342     }
343     break;
344
345 case 7:
346     if (all_number(symbol)){
347         all[top] = symbol;
348         ++top;
349         state = 8;
350     }else if (gap(symbol)){
351         state = 1;
352     }else if (bukv(symbol)){
353         all[top] = symbol;
354         ++top;
355         state = 0;
356     }
357     break;
358 case 777:
359     if (less_number(symbol,arr[6])){
360         all[top] = symbol;
361         ++top;
362         state = 8;
363     }else if (equal_number(symbol, arr[6])){
364         all[top] = symbol;
365         ++top;
366         state = 888;
367     }else if (more_number(symbol, arr[6])){
368         all[top] = symbol;
369         ++top;
370         state = 88;
371     }else if (gap(symbol)){
372         state = 1;
373     }else if (bukv(symbol)){
374         all[top] = symbol;
375         ++top;
376         state = 0;
377     }
378     break;
379 case 77:
380     if (all_number(symbol)){
381         all[top] = symbol;
382         ++top;
383         state = 88;
384     }else if (gap(symbol)){
385         state = 1;
386     }else if (bukv(symbol)){
387         all[top] = symbol;
388         ++top;
389         state = 0;
390     }
391     break;
392
393 case 8:
394     if (all_number(symbol)){
395         all[top] = symbol;
396         ++top;
397         state = 8;
398     }else if (gap(symbol)){
399         state = 1;

```

```

400         }else if (bukv(symbol)){
401             all[top] = symbol;
402             ++top;
403             state = 0;
404         }
405         break;
406     case 888:
407         if (less_number(symbol, arr[7])){
408             all[top] = symbol;
409             ++top;
410             state = 9;
411         }else if (equal_number(symbol, arr[7])){
412             all[top] = symbol;
413             ++top;
414             state = 999;
415         }else if (more_number(symbol, arr[7])){
416             all[top] = symbol;
417             ++top;
418             state = 99;
419         }else if (gap(symbol)){
420             state = 1;
421         }else if (bukv(symbol)){
422             all[top] = symbol;
423             ++top;
424             state = 0;
425         }
426         break;
427     case 88:
428         if (all_number(symbol)){
429             all[top] = symbol;
430             ++top;
431             state = 99;
432         }else if (gap(symbol)){
433             state = 1;
434         }else if (bukv(symbol)){
435             all[top] = symbol;
436             ++top;
437             state = 0;
438         }
439         break;
440
441     case 9:
442         if (all_number(symbol)){
443             all[top] = symbol;
444             ++top;
445             state = 10;
446         }else if (gap(symbol)){
447             state = 1;
448         }else if (bukv(symbol)){
449             all[top] = symbol;
450             ++top;
451             state = 0;
452         }
453         break;
454     case 999:
455         if (less_number(symbol, arr[8])){
456             all[top] = symbol;
457             ++top;
458             state = 10;
459         }else if (equal_number(symbol, arr[8])){
460             all[top] = symbol;
461             ++top;
462             state = 101010;
463         }else if (more_number(symbol, arr[8])){
464             all[top] = symbol;
465             ++top;
466             state = 1010;
467         }else if (gap(symbol)){
468             state = 1;
469         }else if (bukv(symbol)){
470             all[top] = symbol;
471             ++top;
472             state = 0;
473         }
474         break;
475     case 99:
476         if (all_number(symbol)){
477             all[top] = symbol;
478             ++top;
479             state = 1010;
480         }else if (gap(symbol)){
481             state = 1;
482         }else if (bukv(symbol)){
483             all[top] = symbol;
484             ++top;
485             state = 0;
486         }
487         break;
488
489     case 10:
490         if (all_number(symbol)){
491             all[top] = symbol;
492             ++top;
493             state = 11;
494         }else if (gap(symbol)){
495             state = 1;
496         }else if (bukv(symbol)){
497             all[top] = symbol;
498             ++top;
499             state = 0;

```



```

500     }
501     break;
502 case 101010:
503     if (less_number(symbol, arr[9])){
504         all[top] = symbol;
505         ++top;
506         state = 11;
507     }else if (equal_number(symbol, arr[9])){
508         all[top] = symbol;
509         ++top;
510         state = 111111;
511     }else if (more_number(symbol, arr[9])){
512         all[top] = symbol;
513         ++top;
514         state = 12;
515     }else if (gap(symbol)){
516         state = 1;
517     }else if (bukv(symbol)){
518         all[top] = symbol;
519         ++top;
520         state = 0;
521     }
522     break;
523 case 1010:
524     if (all_number(symbol)){
525         all[top] = symbol;
526         ++top;
527         state = 12;
528     }else if (gap(symbol)){
529         state = 1;
530     }else if (bukv(symbol)){
531         all[top] = symbol;
532         ++top;
533         state = 0;
534     }
535     break;
536
537 case 11:
538     if (all_number(symbol)){
539         all[top] = symbol;
540         ++top;
541         state = 12; // a 12 СОСТОЯНИЕ УПОРЯ > MAX_INT
542     }else if (gap(symbol)){
543         //top = top_old;
544         state = 1; // НЕЛЬЗЯ МЕНЬШЕ MAX_INT => НЕДОПОЛНИТЬ К СЛЕДУЮЩЕМУ УПОРЯ
545     }
546     }else if (bukv(symbol)){
547         //top = top_old;
548         all[top] = symbol;
549         ++top;
550         state = 0;
551     }
552     break;
553 case 111111:
554     if (all_number(symbol)){
555         all[top] = symbol;
556         ++top;
557         state = 12; // a 12 СОСТОЯНИЕ УПОРЯ > MAX_INT
558     }else if (gap(symbol)){
559         state = 1; // a 13 СОСТОЯНИЕ УПОРЯ НЕЛЬЗЯ МЕНЬШЕ MAX_INT
560     }else if (bukv(symbol)){
561         all[top] = symbol;
562         ++top;
563         state = 0;
564     }
565     break;
566
567 case 12:
568     if (all_number(symbol)){
569         all[top] = symbol;
570         ++top;
571         state = 12; // a 12 СОСТОЯНИЕ УПОРЯ > MAX_INT
572     }else if (gap(symbol)){
573         printf("%s\n", all);
574         state = 1; // НЕЛЬЗЯ ДОПОЛНИТЬ СЧИСЛО. НЕДОПОЛНИТЬ К 1 СОСТОЯНИЕ. НЕЛЬЗЯ ДОПОЛНИТЬ СЛЕДУЮЩЕМУ УПОРЯ
575     }else if (bukv(symbol)){
576         all[top] = symbol;
577         ++top;
578         state = 0;
579     }
580     break;
581
582     }
583 }
584 return 0;
585 }
586

```

Тест:

```

37 void unit_test(){
38     assert(all_number('2') == 1);
39     assert(all_number('D') == 0);
40     assert(all_number(' ') == 0);
41     assert(all_number('/') == 0);
42     assert(less_number('1', '2') == 1);
43     assert(less_number('6', '2') == 0);
44     assert(less_number('v', '2') == 0);
45     assert(more_number('6', '3') == 1);
46     assert(more_number('0', '5') == 0);
47     assert(more_number('d', '2') == 0);
48     assert(equal_number('4', '4') == 1);
49     assert(equal_number('1', '4') == 0);
50     assert(equal_number('6', '4') == 0);
51     assert(equal_number('h', '4') == 0);
52     assert(equal_number('\t', '4') == 0);
53     assert(gap(' ') == 1);
54     assert(gap("f") == 0);
55     assert(bukv('h') == 1);
56     assert(bukv('0') == 0);
57 }
58
59 int main(void)
60 {
61     unit_test();

```



**9. Дневник отладки** должен содержать дату и время сеансов отладки и основные события (ошибки в сценарии и программе, нестандартные ситуации) и краткие комментарии к ним. В дневнике отладки приводятся сведения об использовании других ЭВМ, существенном участии преподавателя и других лиц в написании и отладке программы.

№ Лаб.	Дата	Время	Событие	Действие по исправлению	Примечание
Дом	31.12.22	16:00	ААаа пытаюсь создать из цифр MAX_INT список((( Сложно. Толи Си ничего не умеет, толи я.	получается, что список получился в обратном порядке( теперь нужно как-то сделать реверс списка, заносив каждый элемент в новый	Грустно, решила, что ничего не умеет ни Си, а я(

#### 10. Замечания автора по существу работы

#### 11. Выводы

В этой лабораторной работе было много боли, пришлось «перелопатить» кучу сайтиков, чтобы понять, как добавлять в си элемент в конец списка, разобраться, как полностью очистить списки (memset ,кстати, очень пригодилась, чтобы каждый раз очищать список, если слово мне не подошло и его не надо выводить). В целом лабораторная мне понравилась, кажется, мы потихоньку находим общий язык с Си ☺

Недочёты при выполнении задания могут быть устранены следующим образом: --

Подпись студента \_\_\_\_\_