

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):**

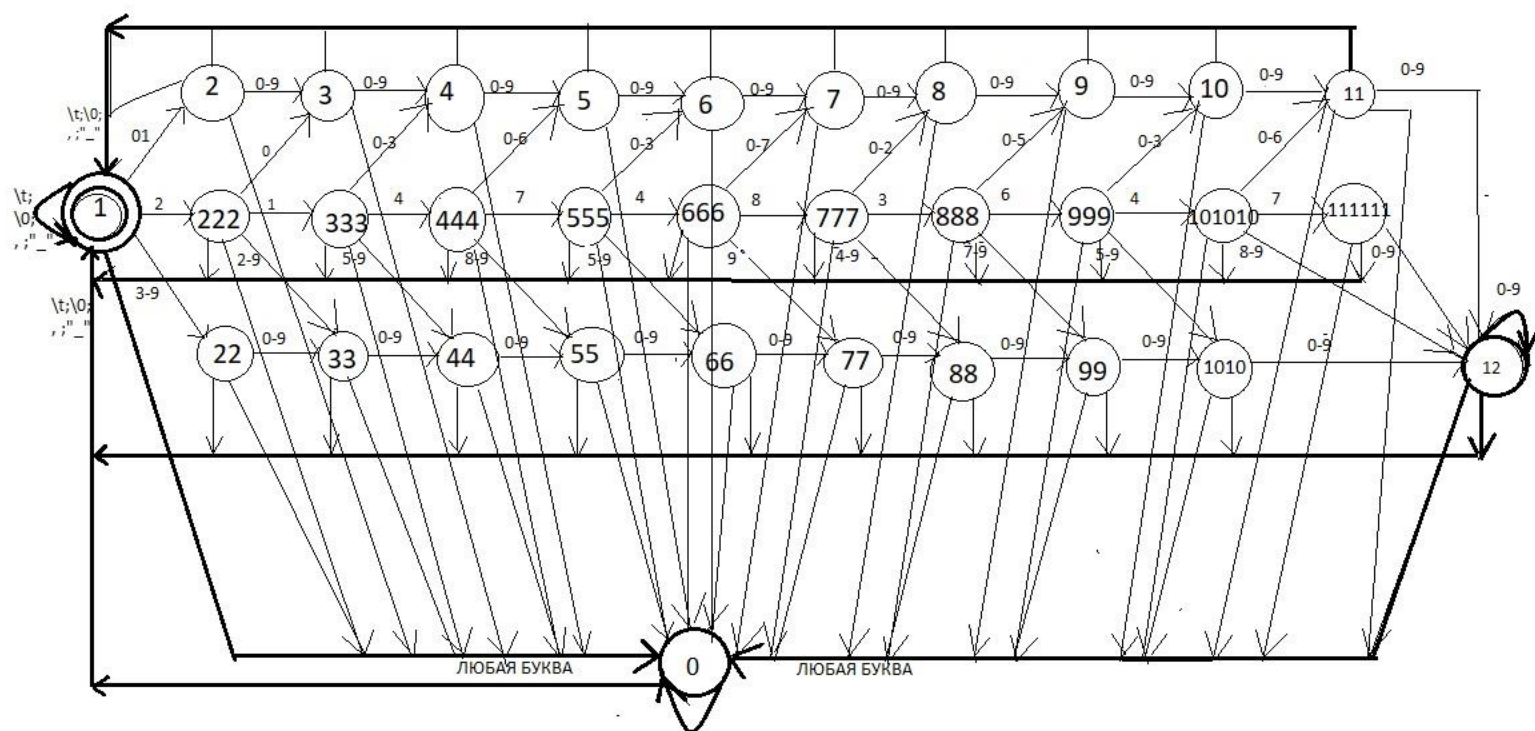
*Процессор AMD Ryzen 5 5500U with Radeon Graphics @ 2.100GHz с ОП 9812 Мб, SSD 512 Гб.
Монитор 1920x1080*

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

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; // находим остаток от деления на 10
48         ++i;
49         int_max /= 10;
50     }
51     for (; i!= 0; --i)
52     {
53         arr[10-i] = tarr[i-1]; // записываем остаток в массив
54     }
55     printf("Input: ");
56
57     int state = 1;
58     int k;
59     char symbol;
60     char all[20] = " "; // массив для хранения символов
61     int top = 0; // топ - указатель на последний элемент массива
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 22:
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 case 3:
153     if (all_number(symbol)){
154         all[top] = symbol;
155         ++top;
156         state = 4;
157     }else if (gap(symbol)){
158         state = 1;
159     }else if (bukv(symbol)){
160         all[top] = symbol;
161         ++top;
162         state = 0;
163     }
164     break;
165 case 333:
166     if (less_number(symbol, arr[2])){
167         all[top] = symbol;
168         ++top;
169         state = 4;
170     }else if (equal_number(symbol, arr[2])){
171         all[top] = symbol;
172         ++top;
173         state = 444;

```

```

174     }
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     case 4:
203         if (all_number(symbol)){
204             all[top] = symbol;
205             ++top;
206             state = 5;
207         }else if (gap(symbol)){
208             state = 1;
209         }else if (bukv(symbol)){
210             all[top] = symbol;
211             ++top;
212             state = 0;
213         }
214         break;
215     case 444:
216         if (less_number(symbol,arr[3])){
217             all[top] = symbol;
218             ++top;
219             state = 5;
220         }else if (equal_number(symbol, arr[3])){
221             all[top] = symbol;
222             ++top;
223             state = 555;
224         }else if (more_number(symbol, arr[3])){
225             all[top] = symbol;
226             ++top;
227             state = 55;
228         }else if (gap(symbol)){
229             state = 1;
230         }else if (bukv(symbol)){
231             all[top] = symbol;
232             ++top;
233             state = 0;
234         }
235         break;
236     case 44:
237         if (all_number(symbol)){
238             all[top] = symbol;
239             ++top;
240             state = 55;
241         }else if (gap(symbol)){
242             state = 1;
243         }else if (bukv(symbol)){
244             all[top] = symbol;
245             ++top;
246             state = 0;
247         }
248         break;
249     case 5:
250         if (all_number(symbol)){
251             all[top] = symbol;
252             ++top;
253             state = 6;
254         }else if (gap(symbol)){
255             state = 1;
256         }else if (bukv(symbol)){
257             all[top] = symbol;
258             ++top;
259             state = 0;
260         }
261         break;
262     case 555:
263         if (less_number(symbol,arr[4])){
264             all[top] = symbol;
265             ++top;
266             state = 6;
267         }else if (equal_number(symbol, arr[4])){
268             all[top] = symbol;
269             ++top;
270             state = 666;
271         }else if (more_number(symbol, arr[4])){
272             all[top] = symbol;
273             ++top;
274             state = 66;
275         }else if (gap(symbol)){
276             state = 1;
277         }else if (bukv(symbol)){
278             all[top] = symbol;
279             ++top;
280             state = 0;
281         }
282         break;
283     case 55:
284         if (all_number(symbol)){
285             all[top] = symbol;
286             ++top;
287             state = 66;
288         }else if (gap(symbol)){
289             state = 1;
290         }else if (bukv(symbol)){
291             all[top] = symbol;
292             ++top;
293             state = 0;
294         }
295         break;
296     case 6:
297         if (all_number(symbol)){
298             all[top] = symbol;
299

```

```

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 = 9;
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 88:
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 89:
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 99:
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 case 11:
537     if (all_number(symbol)){
538         all[top] = symbol;
539         ++top;
540         state = 12; // а 12 состояние больше > max_int
541     }else if (gap(symbol)){
542         //top = top_old;
543         state = 1; // состояние меньше max_int => состояние с меньшим инт
544     }else if (bukv(symbol)){
545         //top = top_old;
546         all[top] = symbol;
547         ++top;
548         state = 0;
549     }
550     break;
551 case 111111:
552     if (all_number(symbol)){
553         all[top] = symbol;
554         ++top;
555         state = 12; // а 12 состояние больше > max_int
556     }else if (gap(symbol)){
557         state = 1; // а 13 состояние больше max_int
558     }else if (bukv(symbol)){
559         all[top] = symbol;
560         ++top;
561         state = 0;
562     }
563     break;
564 case 12:
565     if (all_number(symbol)){
566         all[top] = symbol;
567         ++top;
568         state = 12; // а 12 состояние больше > max_int
569     }else if (gap(symbol)){
570         printf("%s\n", all);
571         state = 1; // число полностью считано, переходим к 1 состоянию, чтобы считать следующее число
572     }else if (bukv(symbol)){
573         all[top] = symbol;
574         ++top;
575         state = 0;
576     }
577     break;
578 }
579 }
580 }
581 }
582 }
583 }
584 return 0;
585 }
586 }

```

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	456765fg 45676567656776567

	567898 fgh56787 , 45678567876545677	fgh56787 45678567876545677
--	--	-------------------------------

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 _____