

Задача по теме 3

«Обработка строк»

Задача

Из входного потока вводится произвольное число текстовых строк; конец ввода – конец файла. Длина каждой строки также произвольна.

Каждая строка представляет собой последовательность слов, разделенных пробельными символами.

Получить новую строку, оставив в исходной ее каждое второе слово.

Строка представлена списком. Должен быть модифицирован исходный список .

Задача

Вариант б)

Каждая строка представлена списком, информационное поле которого содержит символ строки.

Строку результата формировать, модифицируя исходный список.

Все функции, за исключением ввода, работают только со списком. При вводе исходной строки следует возвращать строку, представленную списком.

Структура программы

- Функция main()
- Функция ввода строки-списка произвольной длины
- Функции удаления списка и вывода списка в поток
- Функция формирования результирующей строки
- Функции удаления пробелов, удаления слова, пропуска слова

Структура программы



Объявления

```
#include <stdio.h>
#include <malloc.h>

typedef struct Item {
    char c;
    struct Item *next;
} Item;

int getList (Item **);
void putList(Item *);
Item *deleteList(Item *);

...
```

ФУНКЦИЯ getList()

```
int getList(Item **pptr)
{
    char buf[81], *str;
    Item head = {'*', NULL };
    Item *last = &head;
    int n, rc = 1;
    do{
        n = scanf("%80[^\\n]", buf);
        if(n < 0){
            deleteList(head.next);
            head.next = NULL;
            rc = 0;
            continue;
        }
    }
```

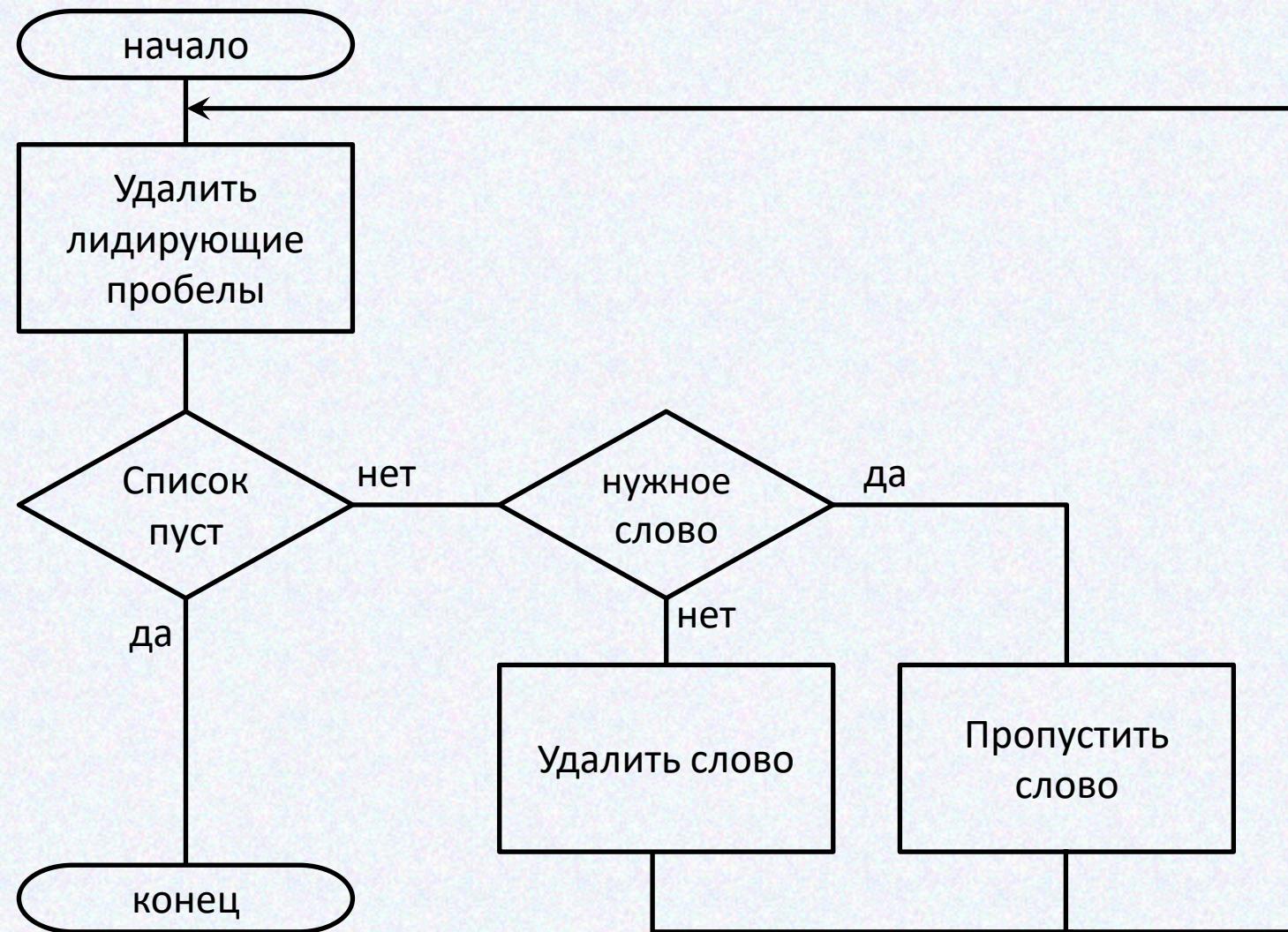
ФУНКЦИЯ getList()

```
if(n > 0){  
    for(str = buf; *str != '\0'; ++str){  
        last->next = (Item *)malloc(sizeof(Item));  
        last = last->next;  
        last->c = *str;  
    }  
    last->next = NULL;  
}  
else  
    scanf("%*c");  
} while(n > 0);  
*pptr = head.next;  
return rc;  
}
```

Тестирование

```
int main()
{
    Item *st;
    while(puts("enter string"), getList(&st)){
        putList("Entered string", st);
        st = deleteList(st);
    }
    return 0;
}
```

Реорганизация списка



Реализация

```
Item *reorg(Item *p)
{
    Item head = {'\0', p},
          *last = &head, *prev = NULL;
    int f = 1;
    while(last &&
          (last->next = delSpace(last->next))) {
        if(f)
            last->next = delWord(last->next);
        f = 0;
    }
}
```

Реализация

```
else{
    prev = skipWord(last->next);
    last = prev->next;
    if( last )
        last->c = ' ';
}
f = !f;
}
```

Реализация

```
if(last && prev){  
    prev->next = NULL;  
    free(last);  
}  
return head.next;  
}
```