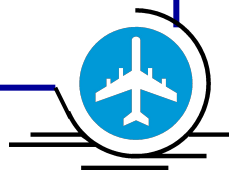
Введение в программирование

Лекция 9.

Примеры символьной обработки



Примеры символьной обработки

- **Задача** 8.1. **Самое длинное слово текста.**
- Входной текст состоит из слов, разделенных пробелами и/или символами "новая строка". Составить программу определения самого длинного слова.

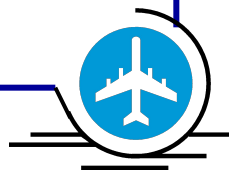
Тест. Вход:

Я снова тут,
Я собран весь <Ctrl-z> <Enter>

Выход:

Самое длинное слово: собран.

Алгоритм символьной обработки разрабатывается, исходя из структуры читаемого им текста, которую удобно описать в виде синтаксических правил - грамматик. Разным грамматикам соответствуют разные варианты алгоритма.



Примеры символьной обработки

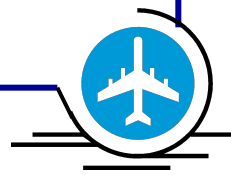
- Решение А. Грамматика текста имеет вид:

текст ::= [слово]...

слово ::= [разделитель]... [символ-слова]...

разделитель ::= пробел | новая-строка

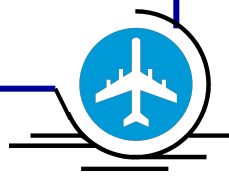
- Каждому знаку повторения "..." синтаксического правила в алгоритме чтения и анализа текста соответствует цикл, знаку "|" (или) – ветвление.
- Структура алгоритма повторяет структуру читаемого текста.
- Алгоритм 8.1а содержит цикл чтения слов, который включает цикл пропуска разделителей и цикл чтения символов слова.



Самое длинное слово

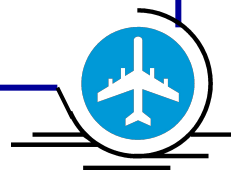
- Обозначим:
  sl - текущее слово,
  dsl - длина текущего слова,
  maxsl - максимальное слово,
  dmaxsl - длина максимального слова.
- Алгоритм 8.1а на псевдокоде:

```
dmaxsl = 0;
while (не конец файла)
{   Пропуск пробелов и "новых строк";
    dsl = 0;
     Чтение текущего слова sl;
     if (dsl > dmaxsl)
              Копирование sl в maxsl; dmaxsl = dsl;
        }
 if (dmaxsl > 0)  Вывод maxsl;
 else
          Вывод "В тексте нет слов";
```



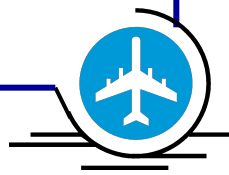
Самое длинное слово

```
/* Программа 8.1а. Слово максимальной длины                    */
    #include <stdio.h>
    #define  DSLMAX  20              /* Максимальная длина слова  */
main ()
    {  char sl [DSLMAX];             /* Текущее слово
       */
       int dsl;                     /* Длина текущего слова      */
   char   maxsl [DSLMAX];           /* Максимальное слово        */
       int dmaxsl;                  /* Длина максимального слова */
   int      sim;                    /*  Текущий символ           */
       int      i;                  /* Текущий индекс копирования*/
   dmaxsl = 0;
```

Самое длинное слово

```
while ((sim = getchar()) != EOF)
     {
    while  (sim==' ' || sim=='\n')   sim=getchar();
    dsl = 0;
    while (sim!=' ' && sim!='\n' && sim!=EOF)
                {sl [dsl++] = sim;       sim = getchar();}
     sl [dsl] = '\0';
     if (dsl > dmaxsl)
          {for (i = 0; maxsl [i] = sl [i]; i++);
            dmaxsl = dsl;
              }
}
if (dmaxsl > 0)      printf ("\nСамое длинное слово: %s\n", maxsl);
else       printf ("\nВ тексте нет слов\n");
return 0;
   }
```

Примеры символьной обработки

- Решение Б. Грамматика текста имеет вид:

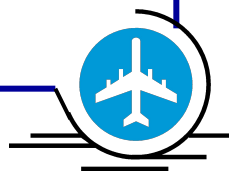
текст ::= символ...

символ ::= разделитель | символ-слова

разделитель ::= пробел | новая-строка | конец-файла

символ-слова - любой символ, кроме разделителей

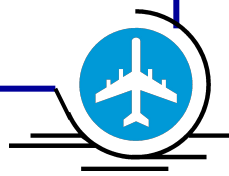
- Алгоритм 8.1б содержит один цикл с постусловием для чтения символов, так как текст содержит хотя бы один символ. В цикле выполняется проверка символа.



Самое длинное слово

- Алгоритм на псевдокоде:

```
dmaxsl = 0; dsl = 0;
do
        { if (текущий символ s - не разделитель)
              текущий символ s в слово sl;
   else                          /* разделитель - конец слова */
       {if (dsl > dmaxsl)
                  {Копирование sl в maxsl;     dmaxsl = dsl;   }
        dsl = 0;
              }
}
        while (не конец файла);
if (dmaxsl > 0)  Вывод maxsl;
 else Вывод "В тексте нет слов";
```

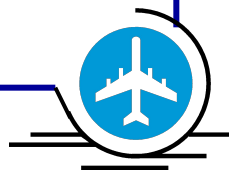


Самое длинное слово

```
/* Программа 8.1б. Слово максимальной длины                    */
    #include <stdio.h>
    #define  DSLMAX  20              /* Максимальная длина слова
     */
void main (void)
{  char sl [DSLMAX];              /* Текущее слово               */
       int      dsl;                 /* Длина текущего слова
     */
   char  maxsl [DSLMAX];          /* Максимальное слово          */
       int      dmaxsl;              /* Длина максимального слова
     */
   int     sim;                   /* Текущий символ              */
       int      i;                   /* Текущий индекс копирования
     */
   dmaxsl = 0; dsl = 0;
```

Самое длинное слово

```
do
    { if ((sim=getchar()) != ' ' && sim!='\n' && sim!=EOF)
          sl [dsl++] = sim;
   else
        { if (dsl > dmaxsl)
        {  sl [dsl] = '\0';
              for (i = 0; maxsl [i] = sl [i]; i++);
           dmaxsl = dsl;
             }
     dsl = 0;
         }
}
    while (sim != EOF);
if (dmaxsl > 0)     printf ("\nСамое длинное слово: %s\n", maxsl);
else     printf ("\nВ тексте нет слов\n");
    }
```



В этом примере отразились весьма характерные и важные закономерности программирования.

1. Существует много решений даже очень простой задачи.
2. По разным критериям лучшими оказываются разные программы, необходим поиск компромиссных вариантов.
3. Выиграешь время - проиграешь память и наоборот: экономя память, увеличишь время решения задачи.
4. Структурное программирование сверху вниз облегчает поиск вариантов алгоритма.
5. Неструктурный алгоритм может оказаться компактнее структурного, но он обычно более запутанный и менее надежный.

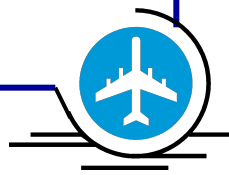
Основные критерии качества программы

1. Надежность.
2. Эффективность по времени.
3. Эффективность по памяти.
4. Удобство эксплуатации.
5. Затраты на разработку программы.

Для уменьшения затрат на разработку важна мобильность программы.

Критерии часто противоречат друг другу.

Особенно характерна закономерность время-память.



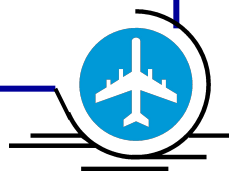
Подпрограмма «Ввод числа»

Пример 9.2. Составить подпрограмму ввода целого числа, перед которым возможны пробелы, символы "новой строки" и знак (подобным образом вводится число по формату %d функции scanf).

Тест. Вход:

12

-5 +234<Ctrl-z><Enter>



Подпрограмма «Ввод числа»

```
/* Программа 9.2.  Ввод целого числа  znach */
void vvod_chisla (int * znach)
{   int sim, znak;
   *znach = 0;  znak = 1;
/*  Пропуск пробелов и "новых строк" до числа          */
   while ((sim=getchar()) == ' ' || sim=='\n');
/* Чтение знака                                        */
   if (sim == '-' || sim == '+')
    {  if (sim == '-') znak = -1;
       sim = getchar();
    }
```

Подпрограмма «Ввод числа»

```
/* Чтение цифр                    */
 while (sim>='0' && sim<='9')
  {  *znach = *znach * 10 + sim - '0';
    sim = getchar();
  }
*znach = *znach * znak;
}
```