# Лабораторная работа 13

Выполните задания 13.1 – 13.3

#### 13.1

Создайте процедуры с параметрами для программы RunBubbleSort из приведенных ниже разделов проекта. На каждом этапе разработки программы соберите одну процедуру и проверьте ее работу. На каждом этапе разработки должна быть рабочая версия BubbleSort. Сформируйте набор тестовых данных, одинаковый для всех этапов.

***a) [#10]***

PROCEDURE CopyFile(VAR InFile, OutFile: TEXT)

Открывайте и закрывайте файлы за пределами процедуры.

***b) [#10]***

PROCEDURE CopyAndSwap(VAR F1, F2: TEXT; VAR Sorted: CHAR)

{ Копируем F1 в F2,проверяя отсортированность

и переставляя первые соседние символы по порядку}

***c) [#10]***

PROCEDURE BubbleSort(VAR InFile, OutFile: TEXT)

Итоговое выполнение:

INPUT:34251  
OUTPUT:12345

INPUT:11523  
OUTPUT:11235

#### 13.2 [#20]

Найдите и исправьте ошибки в процедуре Lexico. Процедура определяет меньшее из 2 слов (слова находятся в первой строке файла, отношение «меньше» определяется как в словаре). Например, строка ABC меньше строки ABD, а строка ABC меньше строки ABCD.

Чтобы запустить процедуру Lexico, считайте первую строку из INPUT в F1, а вторую - в F2.

PROCEDURE Lexico(VAR F1, F2: TEXT; VAR Result: CHAR);

{Result 0, 1, 2 если лексикографический порядок F1 =, <, > чем F2

соответственно. Фактические параметры, соответствующие F1 и F2,

должны быть различными}

VAR

Ch1, Ch2: CHAR;

BEGIN {Lexico}

RESET(F1);

RESET(F2);

Result := '0';

WHILE (NOT EOLN(F1) AND NOT EOLN(F2)) AND (Result = '0')

DO

BEGIN

READ(F1, Ch1);

READ(F2, Ch2);

IF (Ch1 < Ch2)

THEN {Ch1 < Ch2 или F1 короче F2}

Result := '1'

ELSE

IF (Ch1 > Ch2)

THEN {Ch1 > Ch2 или F2 короче F1}

Result := '2'

END {WHILE}

END; {Lexico}

#### 13.3 [#20]

Произведите сборку программы Split из разделов проекта, приведенных ниже. Допишите недостающие разделы DP1.1. и DP1.2.1. Проверьте выполнение.

Выполнение:

INPUT:  
123456  
789

OUTPUT:  
135792468

Может ли какой-либо “Aliasing” произойти в процедуре CopyOut? Если да, приведите пример.

**Разделы проекта для BubbleSort**

DP2

PROGRAM BubbleSort(INPUT, OUTPUT);

{ Сортируем первую строку INPUT в OUTPUT }

VAR

Sorted, Ch, Ch1, Ch2:CHAR;

F1, F2:TEXT;

BEGIN { BubbleSort }

{ Копируем INPUT в F1 }

Sorted :='N';

WHILE Sorted ='N'

DO

BEGIN

{ Копируем F1 в F2,проверяя отсортированность

и переставляя первые соседние символы по порядку}

{ Копируем F2 в F1 }

END;

{ Копируем F1 в OUTPUT }

END.{ BubbleSort }

DP2.1

{Выводим min(Ch1,Ch2) в F2, записывая

отсортированные символы}

IF Ch1 <= Ch2

THEN

BEGIN

WRITE(F2, Ch1);

Ch1:=Ch2

END

ELSE

BEGIN

WRITE(F2, Ch2);

Sorted := 'N'

END

DP2.2

BEGIN { Копируем INPUT в F1 }

REWRITE(F1);

WHILE NOT EOLN

DO

BEGIN

READ(Ch);

WRITE(F1, Ch);

END;

WRITELN(F1)

END;

DP2.3

BEGIN { Копируем F1 в OUTPUT }

.................

(аналогично DP2.2)

END

DP2.4

BEGIN { Копируем F2 в F1 }

.................

(аналогично DP2.2)

END

**Разделы проекта для Split**

DP1

PROGRAM Split(INPUT,OUTPUT);

{Копирует INPUT в OUTPUT,сначала нечетные,а затем четные

элементы}

VAR

Ch,Next: CHAR;

Odds,Evens: TEXT;

{PROCEDURE CopyOut(VAR F1: TEXT; VAR Ch: CHAR);}

BEGIN

{Разделяет INPUT в Odds и Evens}

CopyOut(Odds,Ch);

CopyOut(Evens,Ch);

WRITELN

END.

DP1.1

PROCEDURE CopyOut(VAR F1: TEXT; VAR Ch: CHAR);

BEGIN

{Копируем F1 в OUTPUT}

END;

DP1.2

{Разделяет INPUT в Odds и Evens}

BEGIN

REWRITE(Odds);

REWRITE(Evens);

Next := 'O';

WHILE NOT EOF

DO

BEGIN

WHILE NOT EOLN

DO

{Прочитать Ch, записать в файл, выбранный через

Next,переключить Next}

READLN;

WRITELN(Odds);

WRITELN(Evens)

END;

WRITELN(Odds);

WRITELN(Evens)

END;

DP1.2.1

{Прочитать Ch, записать в файл, выбранный через Next, переключить Next}