**Указатели 1**

**===========**

**var a,b,c:integer; pint: ^integer;**

**begin**

**a:=7; b:=16; pint:=@c; pint^:=a+b;**

**writeln('a=',a,' b=',b',' c=a+b=',c);**

**readln**

**end.**

a b c pint

409000 409002 409004 409006 409008

**409004 409004**

**23**

**16 16**

**7 7 7 77777 7777777 rrr77**

pint c <- в стеке

(stack)

new(pint);

pint <- в куче

(heap)

pint:=@a; память теряется (становится «мусором»)

dispose(pint); - освобождение памяти.

**Связный (цепной) список:**

(однонаправленный линейный связный список)

ы **s3**

ы **s2**

ы **s1**

Lst <- указатель на голову списка.

**type TPItemList=^TItemList;**

**TInfo=string;**

**TItemList= record**

**info: TInfo; {Информационное поле}**

**nxt: TPItemList {Указатель на след. эл-т}**

**end;**

**TList=TPItemList;**

**var Lst: TList;**

**begin Lst:=nil;**

**. . . . . . .**

**Задание Zd01\_1:**

**Используя каркас программы lst1\_nn.pp,**

**реализуйте**

**1) процедуру outputLst(lst: TList)**

**вывода на экран множества слов**

**из списка lst;**

**2) процедуру**

**add2begLst(var lst: TList; x:TInfo)**

**добавления нового слова x в**

**начало списка слов lst.**

ы **s3**

ы **s2**

ы **s1**

Lst pnew

ы **s4**

**Сравните Вашу реализацию со следующей:**

**{--- Вывод элементов списка на экран ---}**

**procedure outputLst(lst: TList);**

**begin**

**while lst<>nil do begin**

**write(lst^.info,' ');**

**lst:=lst^.nxt**

**end;**

**writeln**

**end;**

**{--- Добавление элемента в начало цепного (связного) списка --}**

**procedure add2begLst(var lst: TList; x:TInfo);**

**var pnew:TPItemList;**

**begin**

**new(pnew); pnew^.info:=x;**

**pnew^.nxt:=lst; lst:=pnew**

**end;**

**Задание Zd01\_2 (lst2\_nn.pp):**

**Добавьте в предыдущую программу**

**реализацию процедуры**

**add2endLst(var lst: TList; x:TInfo)**

**добавления нового слова x в конец**

**связного списка lst.**

**Сравните Вашу реализацию со следующей:**

**{--- Добавление элемента в конец цепного (связного) списка --}**

**procedure add2endLst(var lst: TList; x:TInfo);**

**var pnew,p:TPItemList;**

**begin new(pnew); pnew^.info:=x; pnew^.nxt:=nil;**

**if lst=nil then lst:=pnew**

**else begin p:=lst;**

**while p^.nxt<>nil do p:=p^.nxt;**

**p^.nxt:=pnew**

**end;**

**end;**

**Второй способ (использование**

**указателя на указатель):**

**{--- Добавление элемента в конец цепного (связного) списка --}**

**procedure add2endLst(var lst: TList; x:TInfo);**

**var pp:^TPItemList; pnew:TPItemList;**

**begin pp:=@lst;**

**new(pnew); pnew^.info:=x; pnew^.nxt:=nil;**

**while pp^<>nil do pp:=@(pp^^.nxt);**

**pp^:=pnew**

**end;**

**Задание Zd01\_3 (lst3\_nn.pp, lst3m\_nn.pp):**

**Замените все процедуры вставки нового**

**элемента в предыдущей программе на**

**процедуру add2SortLst(var lst: TList; x:TInfo),**

**вставляющую новое слово в лексикографи-**

**чески упорядоченную (как в словаре) после-**

**довательность слов таким образом, чтобы эта**

**упорядоченность сохранялась.**

**Выполните реализацию в 2-х вариантах:**

**а) без использования указателя на указатель**

**(lst3\_nn.pp);**

**б) с использованием указателя на указатель**

**(lst3m\_nn.pp).**