## Лабораторная работа № 20

**Использование указателей**

**Цель работы:**

- приобретение навыков в использовании указателей разных типов;

- приобретение навыков в поиске и исправлении ошибок при работе с указателями разных типов;

- приобретение навыков в работе со связанными и несвязанными динамическими структурами данных.

1. Теоретическая часть

Перед началом выполнения данной работы необходимо изучить раздел 22 «Указатели и ссылочные переменные» конспекта лекций по дисциплине «Языки программирования».

2. Практическая часть

2.1. Задания для выполнения

Задание 1. Пусть имеются следующие объявления:

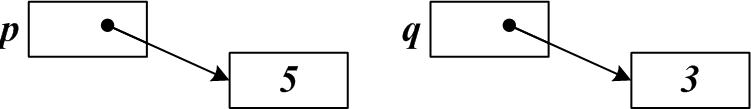
*type*

*rеf = ^integer;*

*var*

*p, q : ref;*

Пусть переменные *р* и *q* имеют значения, соответствующие графическому представлению, показанному на рисунке ниже.



Требуется ответить на следующие вопросы:

1) что является значением переменной *р:* ссылка на объект (переменную) целого типа или сам этот объект;

2) что обозначает *р^:*

- ссылку на объект целого типа,

- сам этот объект,

- целое 5;

3) каковы типы *р* и *p^* ;

4) что будет выдано на экран и какое будет графическое представление размещения значений в памяти (аналогичное показанному на рисунке выше) в результате выполнения *каждого* из действий, обозначенных ниже символами *а, б, в* и *г*, при условии, что все эти действия выполняются строго в указанном порядке сверху вниз и *не* независимые:

*а) p^:= q^;*

*б) if p = q then p:= nil else if p^ = q^ then q: = p;*

*в) if p = q then q^ := 4;*

*г) writeln(p^)?*

Задание 2. Пусть имеются следующие объявления:

*type*

*D = record*

*a : boolean;*

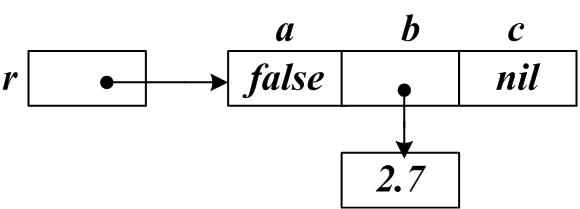
*b, c : ^rеа1*

*end;*

*var*

*r : ^D;*

Пусть ссылочная переменная *r* имеет значение в соответствии с графическим представлением, показанным на рисунке ниже.



Нарисовать, что изменится на приведенном выше рисунке после выполнения *каждого* из действий, обозначенных ниже символами *а, б* и *в*, при условии, что эти действия выполняются строго в указанном порядке сверху вниз и *не* независимые:

*а) if r^.b <> nil*

*then r^.с := r^.b;*

*б) r^.b^ := r^.с^ - 1.4;*

*в) r^.а := r^.b = r^.c .*

Задание 3. Пусть имеются следующие объявления:

*var*

*p, q : ^integer;*

*r : ^char;*

Необходимо определить, есть ли среди приведенных ниже операторов неправильные. Если есть, то почему и как их исправить?

1. *p:= q;*
2. *q:= r;*
3. *p :=nil;*
4. *r := nil;*
5. *q := p^;*
6. *p^ := nil;*
7. *r^ := p^;*
8. *q^ := ord(r^);*
9. *if r <> nil then r^ := nil^;*
10. *if q > nil then q^ := p^;*
11. *if q = p then write(q);*
12. *if q<>r then read(r^).*

Задание 4. Пусть имеется программа следующего вида:

*program dynamic;*

*var*

*x : ^boolean;*

*y : boolean;*

*begin*

*{A} new(x);*

*{B} x^:=true; y:=not x^;*

*{C} dispose(x);*

*{D} writeln(y);*

*end.*

Требуется ответить на следующие вопросы.

1. Какие переменные существуют (видимы) в каждой из точек программы, обозначенных символами *А, В, С* и *D*, и каковы значения переменных в эти моменты?
2. Какими будут графические представления размещения значений объектов программы в памяти (аналогичные показанным на рисунках для заданий 1 или 2), соответствующие каждому из действий *A, B, C* и *D*? Обосновать эти графические представления.
3. Почему объекты (переменные), создаваемые процедурой *new* и уничтожаемые процедурой *dispose,* называют динамическими? Имеют ли такие объекты имя?
4. Можно ли переменной *х* присвоить ссылку на переменную *у*?
5. Можно ли с помощью процедуры *dispose* уничтожить переменные *х* и *у*?

Задание 5. Пусть имеются следующие объявления:

*type*

*A = ^char;*

*В = record*

*f1: char;*

*f2 : A;*

*end;*

*var*

*p: ^B;*

*q: A;*

Нарисовать графическое представление размещения в памяти значений (как в рисунках для вариантов 1 или 2) ссылочных переменных *р* и *q* после выполнения *каждого* из приведенных ниже действий, обозначенных символами *а, б, в, г* и *д*, при условии, что эти действия выполняются строго в указанном порядке сверху вниз и *не* независимые.

а) new(q);

б) *q^ := '7';*

в) *new(p);*

г) *p^.f1 := succ(q^);*

д) *p^.f2 := q.*

Задание 6. Найти и исправить ошибки в приведенной ниже программе. При исправлениях надо попытаться угадать общую логику исходной программы (когда надо выделять память, когда можно разыменовывать, когда можно вводить значения и т.п.) и стремиться к *минимальным* изменениям в замысле и тексте исходной программы.

Нарисовать графическое представление размещения в памяти значений всех переменных (как в рисунках для вариантов 1 или 2):

*program errors;*

*var*

*a, b : ^integer;*

*begin*

*if a = nil*

*then read(a);*

*a^ := 5;*

*b := nil;*

*b^ := 2;*

*new(b); read(b^); writeln (b, b^);*

*new(a); b := a; dispose(a); b^ := 4;*

*end.*

Задание 7. Какие приведенные ниже описания являются недопустимыми? Если такие (недопустимые) описания нашлись, то почему они являются недопустимыми и как их исправить?

Нарисовать графическое представление размещения в памяти значений всех переменных (как в рисунках для вариантов 1 или 2):

*type*

*A = ^0..9;*

*В =record*

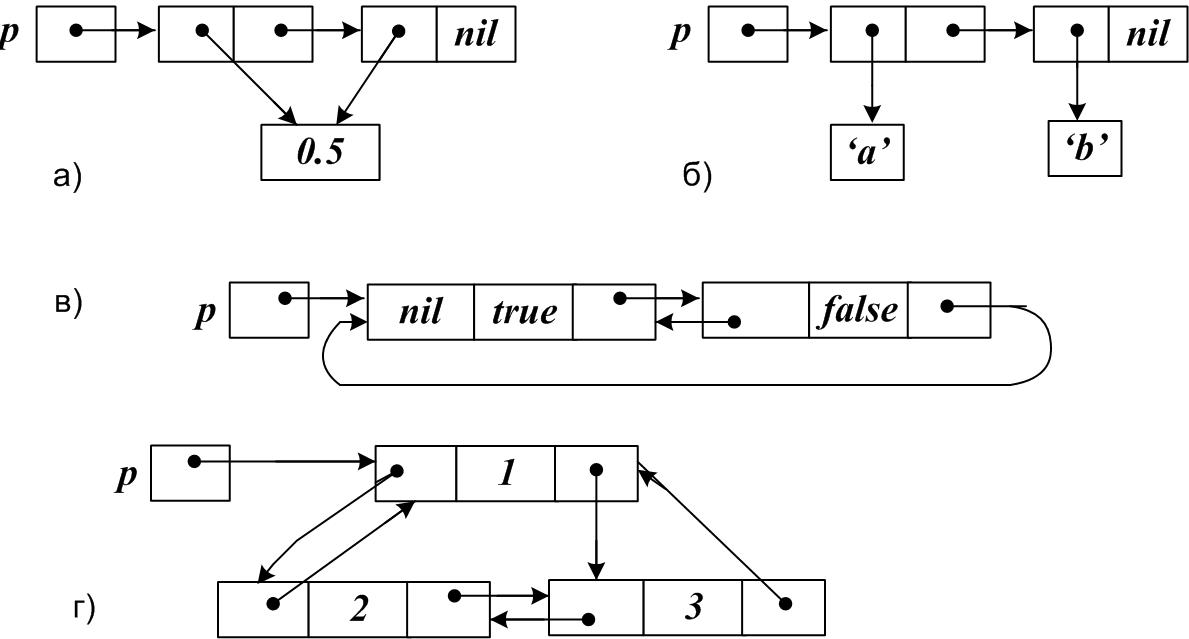
*p : real;*

*q : C;*

*end;*

*C = ^B .*

Задание 8. Требуется для каждого из четырех вариантов, обозначенных на рисунке ниже символами *а, б, в* и *г*, написать программы, в которых описать переменную-указатель *р* и необходимые (в соответствии с рисунком) другие переменные и записать операторы, присваивающие всем переменным и полям записей значения в соответствии с рисунком. Считать, что варианты, обозначенные на рисунке символами *а, б, в* и *г*, являются *независимыми*, а стрелки на рисунке указывают *на весь узел*, а не на отдельные поля записи.



Задание 9. Являются ли допустимыми в языке Паскаль следующие конструкции:

1. *P^[2]*
2. *q^ + 2*
3. *q^[2]^*
4. *r^^ ?*

Для *каждой найденной допустимой конструкции* нарисовать графическое представление размещения в памяти значений переменных (как в рисунках для вариантов 1 или 2) и привести *примеры их использования*.

Задание 10. Пусть имеются следующие объявления:

*type*

*цепочка = ^звено;*

*звено = record*

*элем : integer;*

*след : цепочка;*

*end;*

*var*

*р : цепочка;*

В соответствии с приведенным ниже рисунком для вариантов, обозначенных символами *а, б, в* и *г*, необходимо записать последовательности операторов, которые:

1) преобразуют состояние*,* обозначенное на рисунке символом *а,* к состоянию, обозначенному на рисунке символом *б*;

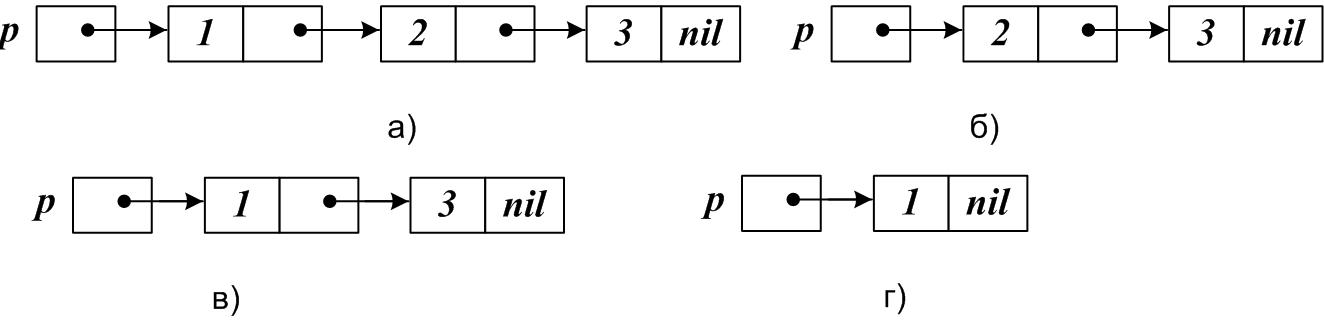
2) преобразуют состояние*,* обозначенное на рисунке символом *а,* к состоянию, обозначенному на рисунке символом *в*;

3) преобразуют состояние*,* обозначенное на рисунке символом *а,* к состоянию, обозначенному на рисунке символом *г*.

Требования к выполнению задания:

- считать, что действия в пунктах 1, 2 и 3 выполняются *независимо друг от друга*;

- удалять звенья из цепочки надо с освобождением занимаемой ими динамической памяти.



Задание 11. Пусть имеются следующие объявления:

*type*

*chain = ^elem;*

*elem = record*

*data : integer;*

*link : chain*

*end;*

*var*

*p, q : chain;*

Требуется нарисовать графическое представление размещения в памяти значений переменных (как в рисунках для вариантов 1 или 2) после выполнения следующих групп операторов:

1. *new(p); p^.data := 4; p^.link := nil;*
2. *new(p); p^.data := 7; p^.link := p;*
3. *new(q); q^. data := 2; q^.link := nil;*

*new(p); p^.data := 1; p^.link := q;*

1. *new(p); p^.data := 5; new(p^.link); p^.link := p .*

Считать, что действия в пунктах 1, 2, 3 и 4 выполняются *независимо* друг от друга.

2.2. Требования к содержанию отчета

Отчет о лабораторной работе должен включать:

1. Конспект теоретической части.

2. Графические представления размещения в памяти значений переменных для каждого задания.

3. Тексты программ с комментариями.

4. Развернутые ответы на все вопросы, поставленные в каждом варианте задания.

2.3. Контрольные вопросы

1. Что такое указатель?
2. Чем отличаются типизированные указатели от нетипизированных?
3. Какие действия можно, а какие нельзя выполнять с нетипизированными указателями?
4. Чем отличаются указатели от ссылок?
5. Как выполняется операция разыменования указателя?
6. Что такое адресная арифметика и где она используется?
7. Что такое динамические переменные, какие они бывают и в чем их отличия от обычных переменных?
8. Как выделять и освобождать память для динамической переменной?
9. Какие возможны ошибки при работе с указателями?
10. Какие известны связанные и несвязанные динамические переменные?