3. ГЕОМЕТРИЧЕСКИЕ ВЫЧИСЛЕНИЯ

В этой главе описаны программы, ориентированные на обработку геометрической информации. Геометрические вычисления широко применяются в машинном проектировании и при подготовке программ для станков с числовым программным управлением (ЧПУ). Графор имеет достаточно полный набор геометрических операций с простейшими геометрическими элементами.

Реализовано два комплекта программ, которые рассматриваются в двух параграфах, составляющих эту главу. В первом случае не предполагается каких-либо иных структур данных для хранения геометрических элементов кроме массива - стандартной структуры фортрана. Во втором случае вводится понятие буфера, состоящего из двух позиций, и определяются операции с этим буфером.

3.1. Геометрические операции

К простейшим геометрическим элементам (примитивам) мы относим точку, прямую и окружность. Точка задается координатами X и Y. Прямая задается коэффициентами A, B, C в уравнении $Ax + By + C = 0$ (где A, B, C могут иметь любые значения, лишь бы коэффициенты A, B не были нулями оба сразу). Окружность задается координатами центра (a, b) и радиусом R и удовлетворяет уравнению: $(x-a)^2 + (y-b)^2 = R^2$.

Геометрические элементы описаны в программах как одномерные массивы: точка - массив из двух вещественных чисел, прямая и окружность - массив из трех вещественных чисел.

Существуют различные способы определения плоских геометрических элементов. Каждому из них соответствует подпрограмма. Описанные ниже программы, конечно, не охватывают всех возможных способов задания геометрических элементов. Реализованы наиболее часто встречающиеся в практике ситуации. Другие способы определения геометрических элементов могут быть реализованы с помощью комбинации нескольких имеющихся в наборе подпрограмм.

Имя подпрограммы однозначно указывает тип определяемого геометрического элемента (первая буква в имени программы) и способ ее параметризации (остальные буквы в имени). Принята английская мнемоника. Например, имя подпрограммы PILC расшифровывается так: определяются точки (P), полученные при пересечении (I) прямой линии (L) с окружностью (C). Имя CTLLP означает, что определяются окружности (C), касающиеся (T) двух прямых линий (LL) и проходящие через заданную точку (P).

Как правило, начальные буквы в наименовании подпрограмм зависят от типов геометрических элементов: P - точка (point); L - прямая (line); C - окружность (circle), A - угловая величина (angle). Имеются отступления от этого правила. В пакете есть несколько программ, в которых определяется не вся окружность, а только ее центр. Первые три буквы в названии этих программ – CNC (CN - сокращение от centre, C - от circle).

Фактические параметры могут быть как входными, так и выходными. Они содержат информацию о форме и положении определяемого геометрического элемента относительно системы координат или относительно других геометрических элементов, определенных ранее. Алгоритмы решения задач не представляют большой сложности. При их реализации использовались методы аналитической геометрии.

Особенностями многих алгоритмов решения геометрических задач является необходимость отбирать нужное решение из нескольких вариантов, удовлетворяющих условию задачи. Например, из двух точек пересечения прямой с окружностью выбрать ту, которая имеет большую или меньшую ординату, либо выбрать точку, ближайшую или наиболее удаленную от некоторой прямой и т. д. В программах, где возможны несколько решений, в качестве выходного параметра выдается число возможных решений. А затем уже программы выбора находят из нескольких возможных вариантов, удовлетворяющих условию задачи, нужное решение.

В программах, реализующих конкретные геометрические операции, и в программах, выбирающих решения, используются общие блоки. Таких блоков два: GFGMP(4) и GFGMC(16). Геометрические элементы почти во всех программах записаны в блок в произвольном порядке. Только в программах CNCTCL и CNCTCC сначала записываются окружности, касающиеся снаружи окружности, стоящей первой в списке параметров программ, а затем окружности, касающиеся внутри.

Если пользователя не устраивают те критерии, по которым производится выбор решения, то он может написать свои программы выбора решения с использованием другого критерия. Для этого есть вся необходимая информация: известны количество решений, общий блок, куда записаны решения, представления геометрических элементов (точка - *X*, *Y*, прямая - *A*, *B*, *C*, окружность - *a*, *b*, *R*).

В случаях некорректного обращения происходит выход из данной программы. Если в программе должны быть присвоены какие-то значения некоторым переменным, то при некорректном обращении этого не происходит.

Для каждой из описанных далее программ на рис. 3.1 дается геометрическая иллюстрация. На рисунке для некоторых программ указаны значения признака выбора решения J.

3.1.1. Программы определения точки.

P1. Программа PXY(X,Y, P) определяет точку с координатами X и Y. Параметры программы:

X, Y - координаты точки;

P - полученная точка.

P2. Программа PRAP(P1, R, ALPHA, P2) задает точку в полярных координатах. Параметры программы:

P1 - полюс;

R - полярный радиус;

ALPHA - полярный угол (в градусах);

P2 - полученная точка.

P3. Программа PCNTRC(C, P) задает точку как центр окружности. Параметры программы:

C - окружность;

P - полученная точка.

P4. Программа PAC(C, ALPHA, P) находит точку на заданной окружности, если звестен угол наклона к оси *X*, под которым находится точка. Параметры программы:

C - окружность;

ALPHA - угол (в градусах);

P - полученная точка.

P5. Программа PCNAP(PC, P1, ALPHA, P) определяет на окружности с заданным центром точку, находящуюся на указанном угловом расстоянии от данной точки. Параметры программы:

PC - центр окружности;

P1 - заданная точка на окружности;

ALPHA - угол (в градусах);

P - полученная точка.

Если ALPHA > 0, то точка лежит в направлении против часовой стрелки от заданной точки, если ALPHA < 0 - по часовой стрелке.

P6. Программа PMIDPP(P1, P2, P) определяет точку, находящуюся посередине между двумя заданными точками. Параметры программы:

P1, P2 - заданные точки;

P - полученная точка.

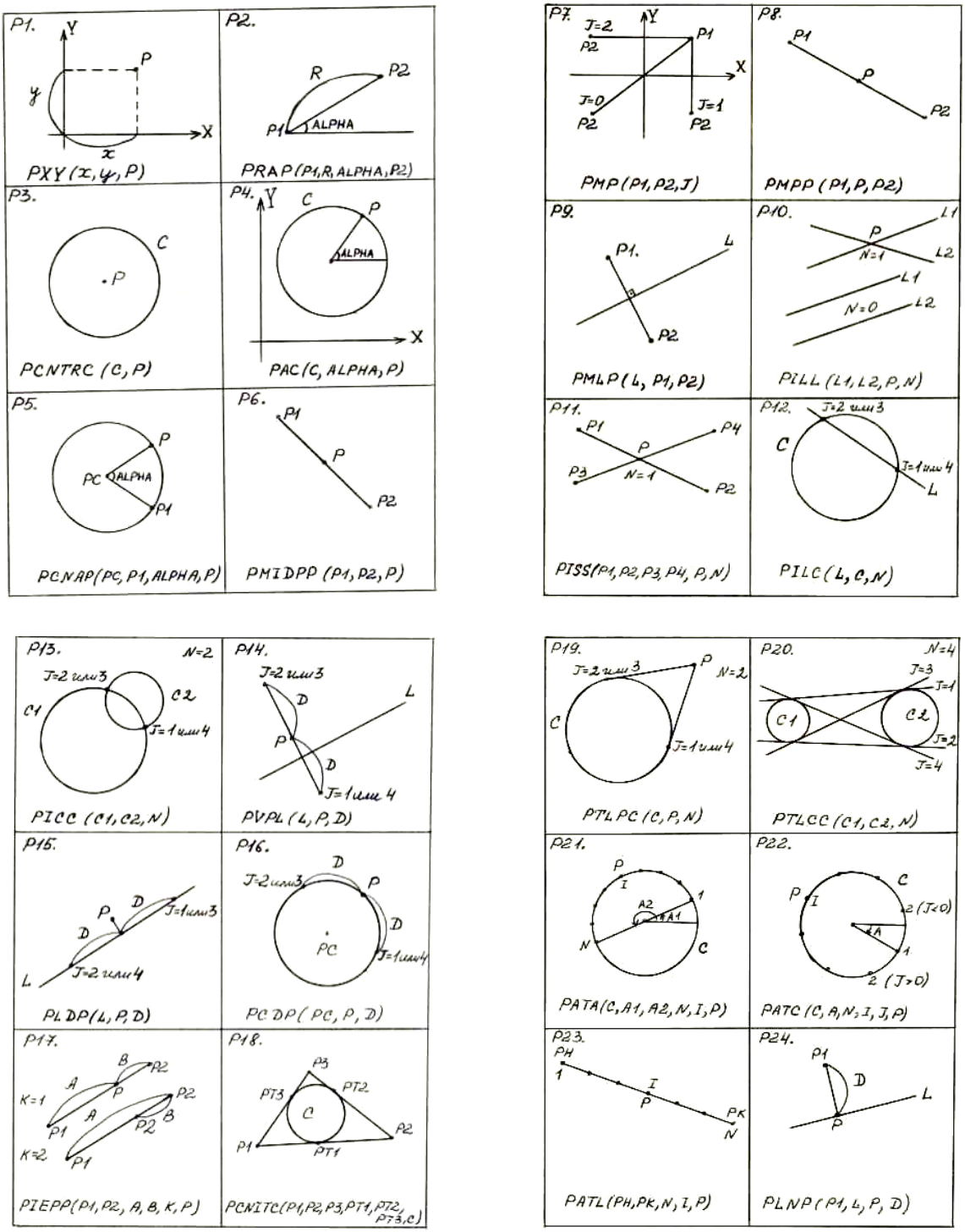

Рис. 3.1. Иллюстрации к программам геометрических вычислений.

P7. Программа PMP(P1, P2, J) находит точку, симметричную заданной относительно начала координат или осей. Параметры программы:

P1, P2 - заданная и полученная точки;

J - признак выбора симметричной точки: J = 0 - по отношению к началу координат, J = 1 - по отношению к оси *X*, J = 2 – по отношению к оси *Y*.

P8. Программа PMPP(P1, P, P2) находит точку, симметричную заданной относительно другой известной точки. Параметры программы:

P1, P2 - заданная и полученная точки;

P - точка, относительно которой надо получить симметричную.

P9. Программа PMLP(L, P1, P2) определяет точку, симметричную заданной точке относительно известной прямой линии. Параметры программы:

L - заданная прямая;

P1, P2 - заданная и полученная точки.

P10. Программа PILL(L1, L2, P, N) определяет точку пересечения двух заданных прямых линий. Параметры программы:

L1, L2 - заданные прямые;

P - полученная точка;

N - параметр, характеризующий число точек: N = 0 - нет точек пересечения (прямые параллельны), N = 1 - одна точка пересечения.

P11. Программа PISS(P1, P2, P3, P4, P, N) определяет точку пересечения двух прямых, заданных отрезками. Параметры программы:

P1, P2 - концевые точки первого отрезка;

P3, P4 - концевые точки второго отрезка;

P - точка пересечения;

N - число точек (параметр N такой же, как в программе PILL).

P12. Программа PILC(L,C, N) находит точки пересечения заданных прямой линии и окружности. Параметры программы:

L, C - заданные прямая и окружность;

N - число точек пересечения: N = 0 - нет общих точек, N = 1 - прямая и окружность касаются, N = 2 - прямая и окружность пересекаются.

Результат в общем блоке GFGMP. Решение выбирается с помощью программы SORTP или программы SPNP.

P13. Программа PICC(C1,C2, N) находит точки пересечения двух заданных окружностей. Параметры программы:

C1, C2 - заданные окружности;

N - число точек: N = 0 - нет общих точек, N = 1 – окружности касаются, N = 2 - окружности пересекаются.

Результат в общем блоке GFGMP. Решение выбирается с помощью программы SORTP или программы SPNP.

P14. Программа PVPL(L,P,D) находит точки, расположенные на указанном расстоянии от заданной точки и лежащие на прямой, проходящей через эту точку перпендикулярно некоторой известной прямой. Параметры программы:

L, P - заданные прямая и точка;

D - расстояние.

Результат в общем блоке GFGMP. Решение выбирается программой SORTP или программой SPNP.

P15. Программа PLDP(L, P, D) находит точки на прямой линии, равноудаленные от заданной точки и расположенные на заданном расстоянии от основания перпендикуляра, опущенного из этой точки на прямую. Параметры программы:

L, P - заданные прямая и точка;

D - расстояние.

Результат в общем блоке GFGMP. Решение выбирается с помощью программы SORTP или программы SPNP.

P16. Программа PCDP(PC, P, D) находит точки на окружности с заданным центром на расстоянии D по дуге от известной точки на той же окружности. Параметры программы:

PC - центр окружности;

P - заданная точка;

D - длина дуги.

Результат в общем блоке GFGMP. Решение выбирается с помощью программы SORTP или программы SPNP.

Если заданная точка совпадает с центром окружности, то обращение к программе считается некорректным.

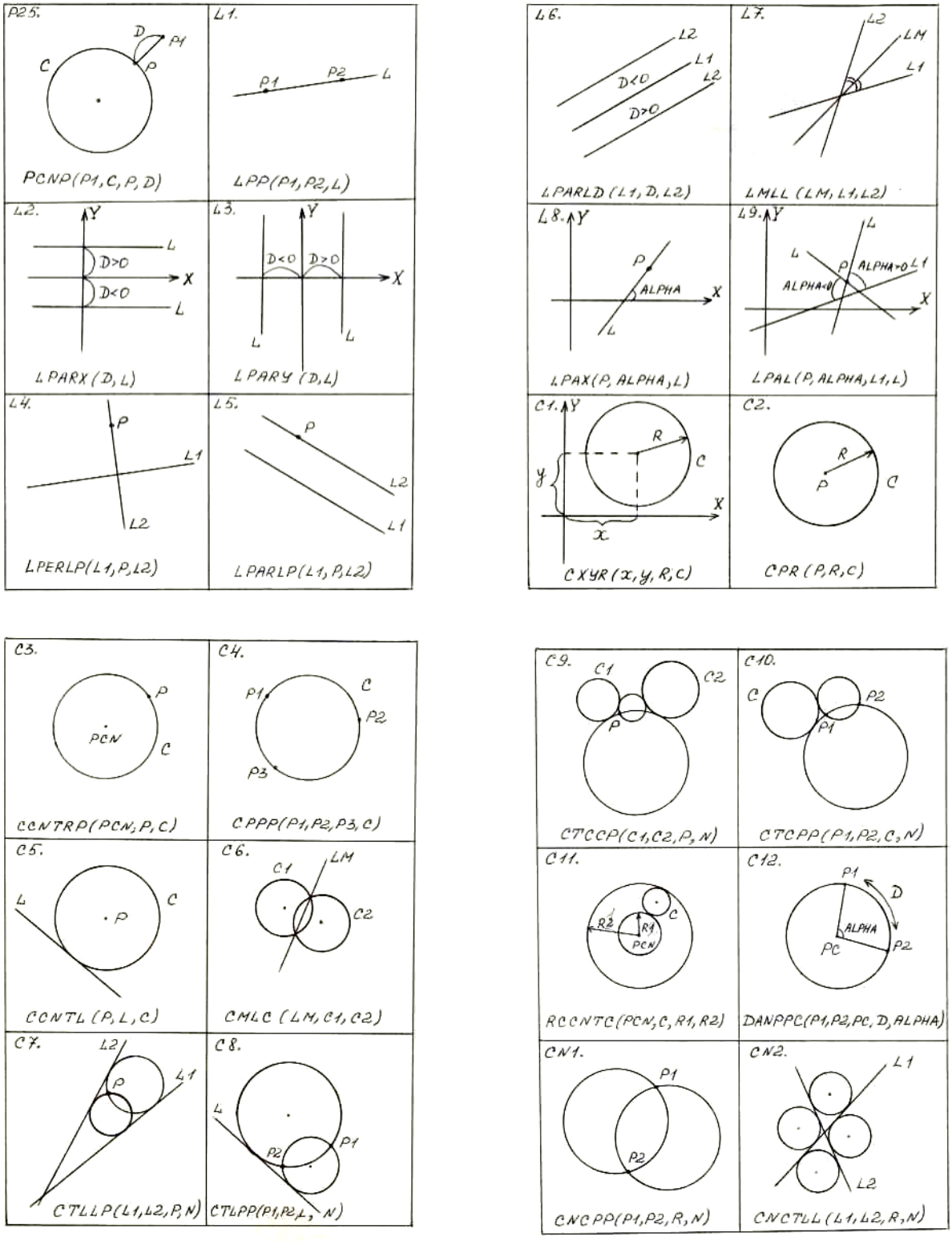

Рис. 3.1. Иллюстрации к программам геометрических вычислений.

P17. Программа PIEPP(P1, P2, A,B, K, P) определяет точку, лежащую на той же прямой, что и две другие заданные точки, и отстоящую от них в заданном отношении A/B. Параметры программы:

P1, P2 - заданные точки;

A/B - заданное отношение;

K - признак расположения точки: K = 1 - точка лежит между заданными точками,

K = 2 - точка лежит вне отрезка, образованного заданными точками;

P - искомая точка.

При A = B, K = 2 обращение к программе считается некорректным.

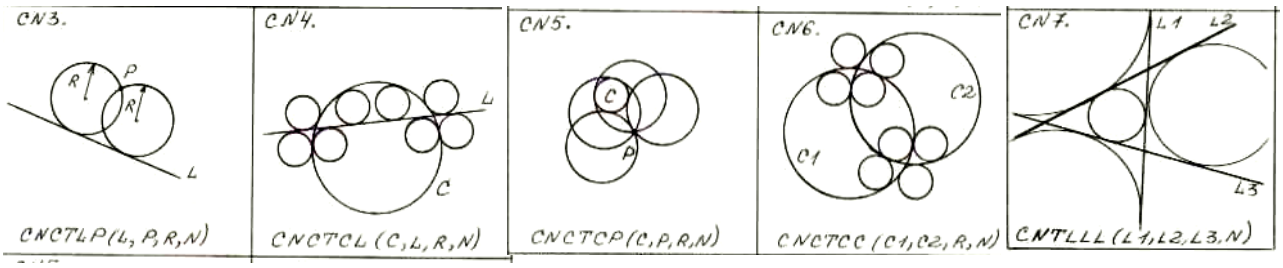

Рис. 3.1. Иллюстрации к программам геометрических вычислений.

P18. Программа PCNITC(P1, P2, P3, PT1, PT2, PT3, C) определяет точки касания окружности, вписанной в треугольник с заданными вершинами, а также вписанную окружность. Параметры программы:

P1, P2, P3 - вершины треугольника;

PT1, PT2, PT3 - точки касания окружности со сторонами треугольника, проходящими соответственно через точки P1 и P2, P2 и P3, P1 и P3;

C - вписанная окружность.

Обращение к программе считается некорректным, если три точки лежат на одной прямой.

P19. Программа PTLPC(C, P, N) определяет точки касания окружности с прямой, проходящей через заданную точку. Параметры программы:

C, P - заданные окружность и точка;

N - число точек касания ($0 \le N \le 2$).

Результат в общем блоке GFGMP. Решение выбирается с помощью программы SORTP или программы SPNP.

P20. Программа PTLCC(C1, C2, N) определяет точки касания прямой с двумя заданными окружностями. Параметры программы:

C1, C2 - заданные окружности;

N - число возможных касательных прямых ($N \le 4$).

Результат в общем блоке GFGMC. Решение выбирается с помощью программы SORTPT. В блок GFGMC записываются сначала две пары точек для внешних касательных, затем две пары точек для внутренних касательных (для каждой пары точек первой в блок заносится точка касания с окружностью, стоящей первой в списке параметров, второй - со второй окружностью).

P21. Программа PATA(C, A1, A2, N, I, P) определяет точку с заданным номером I на дуге окружности, разделенной N точками на равные части. Точки нумеруются от 1 до N. Параметры программы:

C - заданная окружность;

A1, A2 - начало и конец дуги (в градусах);

N - число точек на дуге;

I - номер точки, которую нужно найти;

P - найденная точка.

Обращение к программе считается некорректным, если N < I.

P22. Программа PATC(C, N, A, I, J, P) определяет точку с заданным номером на разделенной на N частей окружности. Параметры программы:

C - окружность;

N - число точек на окружности;

A - угол к оси *X*, под которым находится на окружности первая точка (задан в градусах);

I - номер точки, которую требуется найти;

J - направление нумерации точек: J > 0 - по часовой стрелке, J < 0 - против часовой стрелки;

P - полученная точка.

Обращение к программе считается некорректным, если N < I.

P23. Программа PATL(PH, PK, N, I, P) выбирает точку с заданным номером I на заданном отрезке, разделенном N точками на равные части. Точки нумеруются от 1 до N. Параметры программы:

PH, PK - начальная и конечная точки отрезка;

N - число точек на отрезке;

I - номер точки, которую надо найти;

P - выбранная точка.

Точки нумеруются от начальной к конечной.

Обращение к программе считается некорректным, если N < I.

P24. Программа PLNP(P1, L, P, D) находит точку на заданной прямой, ближайшую к некоторой известной точке, и определяет расстояние между ними. Параметры программы:

P1 - заданная точка;

L - заданная прямая;

P - искомая точка;

D - расстояние между точками.

Если заданная точка лежит на прямой, то искомая точка совпадает с заданной.

P25. Программа PCNP(P1, C, P, D) находит точку на заданной окружности, ближайшую к некоторой известной точке, и определяет расстояние между ними. Параметры программы:

P1 - заданная точка;

C - заданная окружность;

P - искомая точка;

D - расстояние между точками.

Если заданная точка лежит на окружности, то искомая точка совпадает с заданной.

Обращение к программе считается некорректным, если заданная точка совпадает с центром окружности.

3.1.2. Программы определения прямой.

L1. Программа LPP(P1, P2, L) определяет прямую линию (коэффициенты A, B, C), проходящую через две заданные точки. Параметры программы:

P1, P2 - заданные точки;

L - полученная прямая.

L2. Программа LPARX(D, L) определяет прямую линию, параллельную оси абсцисс на заданном расстоянии. Параметры программы:

|D| - расстояние: D > 0 - прямая над осью *X*, D < 0 – прямая под осью *X*;

L - найденная прямая.

L3. Программа LPARY(D, L) определяет прямую линию, параллельную оси ординат на заданном расстоянии. Параметры программы:

|D| - расстояние: D > 0 - прямая справа от оси *Y*, D < 0 - прямая слева от оси *Y*;

L - найденная прямая.

L4. Программа LPERLP(L1, P, L2) определяет прямую линию, проходящую через заданную точку и перпендикулярную заданной прямой. Параметры программы:

L1 - заданная прямая;

P - заданная точка;

L2 - полученная прямая.

L5. Программа LPARLP(L1, P, L2) определяет прямую линию, проходящую через заданную точку и параллельную заданной прямой. Параметры программы:

L1 - заданная прямая;

P - известная точка;

L2 - найденная прямая.

L6. Программа LPARLD(L1, D, L2) определяет прямую линию, параллельную заданной прямой и расположенную на заданном расстоянии. Параметры программы:

L1 - заданная прямая;

|D| - заданное расстояние: D > 0 - прямая проводится в сторону возрастания координаты *X*; D < 0 - в сторону убывания координаты *X*;

L2 - полученная прямая.

В случае, когда заданная прямая линия горизонтальная, то при D > 0 параллельная прямая проводится выше заданной прямой, при D < 0 - ниже.

L7. Программа LMLL(LM, L1, L2) определяет прямую линию, симметричную заданной прямой относительно другой прямой. Параметры программы:

LM - прямая, являющаяся осью симметрии;

L1 - заданная прямая;

L2 - найденная симметричная прямая.

L8. Программа LPAX(P, ALPHA, L) находит прямую линию, проходящую через заданную точку под заданным углом к оси *X*. Параметры программы:

P - заданная точка;

ALPHA - угол (в градусах);

L - найденная прямая.

L9. Программа LPAL(P, ALPHA, L1, L) определяет прямую линию, проходящую через заданную точку и составляющую с заданной прямой известный угол. Параметры программы:

P - точка;

ALPHA - угол (в градусах);

L1 - заданная прямая;

L - найденная прямая.

Если ALPHA > 0, то искомая прямая образуется поворотом на заданный угол относительно заданной прямой против часовой стрелки, если ALPHA < 0 - по часовой стрелке.

3.1.3. Программы определения окружности.

C1. Программа CXYR(X, Y, R, C) определяет окружность по заданным координатам центра и радиусу. Параметры программы:

X, Y - координаты центра;

R - радиус;

C - найденная окружность.

C2. Программа CPR(P, R, C) определяет окружность по заданным точке (центру) и радиусу. Параметры программы:

P, R - заданные точка и радиус;

C - найденная окружность.

C3. Программа CCNTRP(PCN, P, C) определяет окружность по заданным центру и точке, лежащей на окружности. Параметры программы:

PCN - центр окружности;

P - заданная точка;

C - найденная окружность.

C4. Программа CPPP(P1, P2, P3, C) находит окружность, проходящую через три заданные точки, не лежащие на одной прямой. Параметры программы:

P1, P2, P3 - три заданные точки;

C - найденная окружность.

Обращение к программе считается некорректным, если все три точки лежат на одной прямой.

C5. Программа CCNTL(P, L, C) определяет окружность по заданным центру и касательной к окружности. Ее параметры:

P - центр окружности;

L - касательная линия;

C - найденная окружность.

C6. Программа CMLC(LM, C1, C2) определяет окружность, симметричную заданной окружности относительно известной прямой. Параметры программы:

LM - прямая, являющаяся осью симметрии;

C1 - заданная окружность;

C2 - найденная окружность.

C7. Программа CTLLP(L1, L2, P, N) определяет окружности, касающиеся двух заданных прямых и проходящие через заданную точку. Параметры программы:

L1, L2 - заданные прямые;

P - заданная точка;

N - число окружностей ($0 \le N \le 2$).

Результат в блоке GFGMC. Решение выбирается с помощью программы SORTC.

C8. Программа CTLPP(P1, P2, L, N) находит окружности, проходящие через две заданные точки и касающиеся заданной прямой. Параметры программы:

P1, P2 - заданные точки;

L - заданная прямая;

N - число окружностей ($0 \le N \le 2$).

Результат в блоке GFGMC. Решение выбирается с помощью программы SORTC.

C9. Программа CTCCP(C1, C2, P, N) определяет окружности, касающиеся двух заданных окружностей и проходящие через известную точку. Параметры программы:

C1, C2 - заданные окружности;

P - заданная точка;

N - число окружностей ($0 \le N \le 2$).

Результат в блоке GFGMC. Решение выбирается программой SORTC.

C10. Программа CTCPP(P1, P2, C, N) находит окружности, проходящие через две заданные точки и касающиеся заданной окружности. Параметры программы:

P1, P2 - заданные точки;

C - заданная окружность;

N - число окружностей ($0 \le N \le 2$).

Результат в общем блоке GFGMC. Решения выбираетсяется программой SORTC.

C11. Программа RCCNTC(PCN, C, R1, R2) определяет радиусы окружностей с заданным центром, касающихся другой известной окружности. Параметры программы:

PCN - центр окружности;

C - заданная окружность;

R1, R2 - радиусы (R1 - больший радиус, R2 - меньший).

C12. Программа DANPPC(P1, P2, PC, D, ALPHA) находит длину меньшей дуги между двумя заданными точками окружности и ее угловую величину. Параметры программы:

P1, P2 - заданные точки, лежащие на окружности;

PC - центр окружности;

D - длина дуги;

ALPHA - угол (в градусах).

3.1.4. Определение центров окружностей заданного радиуса.

CN1. Программа CNCPP(P1, P2, R, N) находит центры окружностей заданного радиуса, проходящих через две заданные точки. Ее параметры:

P1, P2 - заданные точки;

R - радиус;

N - число окружностей ($0 \le N \le 2$).

Результат в общем блоке GFGMP. Решение выбирается с помощью программы SORTP или программы SPNP.

CN2. Программа CNCTLL(L1, L2, R, N) определяет центры окружностей заданного радиуса, касающихся двух заданных прямых. Параметры программы:

L1, L2 - заданные прямые;

R - радиус;

N - число окружностей (N = 4, если прямые пересекаются, и N = 0, если прямые параллельны).

Результат в общем блоке GFGMC. Решение выбирается с помощью программы SORTCN. Обращение к программе считается некорректным, если прямые параллельны (в том числе совпадают).

CN3. Программа CNCTLP(L, P, R, N) находит центры окружностей заданного радиуса, проходящих через заданную точку и касающихся заданной прямой. Параметры программы:

L, P - заданные прямая и точка;

R - радиус;

N - число окружностей ($0 \le N \le 2$).

Результат в общем блоке GFGMP. Решение выбирается с помощью программы SORTP или программы SPNP.

CN4. Программа CNCTCL(C, L, R, N) определяет центры окружностей заданного радиуса, касающихся известных прямой и окружности. Параметры программы:

C, L - заданные окружность и прямая;

R - радиус;

N - число окружностей ($0 \le N \le 8$).

Результат в общем блоке GFGMC. Решение выбирается с помощью программы SORTCN. (При N = 2 можно воспользоваться программой SORTP или программой SPNP.)

CN5. Программа CNCTCP(C, P, R, N) определяет центры окружностей заданного радиуса, проходящих через заданную точку и касающихся заданной окружности. Параметры программы:

C, P - заданные окружность и точка;

R - радиус;

N - число окружностей ($0 \le N \le 4$).

Результат в общем блоке GFGMC. Решение выбирается с помощью программы SORTCN. (При N = 2 можно воспользоваться программой SORTP или программой SPNP.)

CN6. Программа CNCTCC(C1, C2, R, N) находит центры окружностей заданного радиуса, касающихся двух заданных окружностей. Параметры программы:

C1, C2 - заданные окружности;

R - радиус;

N - число окружностей ($0 \le N \le 8$).

Результат в общем блоке GFGMC. Решение выбирается с помощью программы SORTCN. (При N = 2 можно воспользоваться программой SORTP или программой SPNP.)

CN7. Программа CNTLLL(L1, L2, L3, N) определяет центры окружностей, касающихся трех заданных прямых. Параметры программы:

L1, L2, L3 - заданные прямые;

N - число окружностей (N = 0, 2, 4).

Если прямые параллельны или пересекаются в одной точке, то обращение к программе считается некорректным.

Результат в общем блоке GFGMC. Решение выбирается с помощью программы SORTL.

3.1.5. Программы выбора решения. В тех случаях, когда при выполнении геометрической операции получается несколько решений, результат операции заносится в один из общих блоков. Таких блоков два: GFGMP(4) и GFGMC(16). Программы SORTP и SPNP ведут поиск решения в общем блоке GFGMP, программы SORTC, SORTCN, SORTL и SORTPT - в общем блоке GFGMC.

Программа SORTP(J, P) позволяет выбрать точку из набора, полученного в результате геометрических вычислений с помощью любой из программ PILC, PICC, PVPL, PLDP, PCDP, PTLPC, CNCPP, CNCTLP. Параметры программы:

J - признак выбора точки: J = 1 - с большей *X*-координатой (если *X*-координаты равны, то с большей *Y*-координатой), J = 2 – с меньшей *X*-координатой (если *X*-координаты равны, то с меньшей *Y*-координатой), J = 3 - с большей *Y*-координатой (если *Y*-координаты равны, то

с большей *X*-координатой), J = 4 - с меньшей *Y*-координатой (если *Y*-координаты равны, то с меньшей *X*-координатой);

P - найденная точка.

Программа SORTPT(J, PT1, PT2) предназначена для программы PTLCC. Параметры программы:

J - признак, по которому выбирается решение: J = 1 – обе точки касания находятся слева от линии центров, J = 2 - обе точки касания справа от линии центров, J = 3 - первая точка касания слева, а вторая - справа от линии центров, J = 4 - первая точка касания справа, а вторая - слева от линии центров;

PT1, PT2 - выбранные точки касания.

Линия центров направлена от центра окружности, записанной первой в списке параметров программы PTLCC, к центру окружности, записанной второй. PT1 - точка касания прямой с окружностью, записанной первой в списке параметров программы PTLCC, PT2 – точка касания с окружностью, записанной второй.

Программа SORTC(J, C) позволяет выбрать окружность из набора, полученного в результате геометрических вычислений с помощью любой из программ CTLLP, CTLPP, CTCCP, CTCPP. Выбор окружности определяется положением ее центра. Параметры программы:

J - признак выбора нужной окружности:
J = 1 - с большей *X*-координатой центра (если *X*-координаты равны, то с большей *Y*-координатой),
J = 2 - с меньшей *X*-координатой центра (если *X*-координаты равны, то с меньшей *Y*-координатой),
J = 3 - с большей *Y*-координатой центра (если *Y*-координаты равны, то с большей *X*-координатой),
J = 4 - с меньшей *Y*-координатой центра (если *Y*-координаты равны, то с меньшей *X*-координатой);

C - выбранная окружность.

Программа SORTCN(M, J, PC) предназначена для программ CNCTCC, CNCTCL, CNCTLL, CNCTCP. Параметры программы:

M - признак выбора группы окружностей (нужен для программ CNCTCC и CNCTCL): M = 1 - окружности, касающиеся окружности, записанной первой в списке параметров в программах CNCTCC и CNCTCL, снаружи, M = 1 - касающиеся внутри;

J - признак выбора нужного решения: J = 1 - выбор центра окружности с большей координатой *X*, J = 2 - со второй координатой *X*, J = 3 - с третьей координатой *X*, J = 4 - с наименьшей координатой *X*, J = 5 - с большей координатой *Y*, J = 6 - со второй координатой *Y*, J = 7 - с третьей координатой Y, J = 8 – с наименьшей *Y*-координатой;

PC - выбранный центр окружности.

Программа SORTCN использует служебную программу SORT3.

Замечание. Для всех программ, кроме CNCTCC и CNCTCL, параметр M всегда должен быть равен 1. .

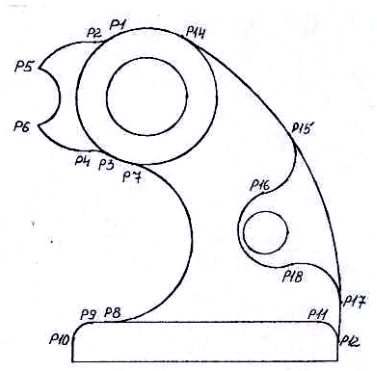

Рис. 3.2. Пример использования программ геометрических вычислений.

Программа SORTL(L, J, C) предназначена для программы CNTLLL. Параметры программы:

L - любая из прямых, определенных в списке параметров программы CNTLLL;

J - признак выбора решения (тот же, что в программе SORTCN);

C - выбранная окружность

Программа SPNP(P, PN) выбирает из двух точек точку, ближайшую к некоторой известной точке.

Параметры программы:

P - заданная точка

PN - искомая точка.

Если расстояния от обеих точек до заданной точки равны, то точка выбирается произвольно.

3.1.6. Примеры. На рис.3.2 показан пример использования программ геометрических вычислений. Для данного чертежа приводится фрагмент программы, представляющий ее вычислительную часть. Рисование осуществлялось с помощью программ Графора.

```
      REAL P1(2), P2(2), P3(2), P4(2), P5(2)
      REAL P6(2), P7(2), P8(2), P9(2)
      REAL P10(2), P11(2), P12(2), P14(2)
      REAL P15(2), P16(2), P17(2), P18(2)
      REAL L1(3), L2(3), L3(3), C1(3)
      REAL C2(3), C3(3), C4(3), C5(3), C6(3)
      REAL C7(3), C8(3), C9(3), C10(3)
      DATA C1/0., 0., 35./,C2/-30., 0., 28./
      DATA C4/-58., 0., 15./
C....ВЫЧИСЛИТЕЛЬНАЯ ЧАСТЬ
      CALL CNCTCC(C1, C2, 15., N)
      CALL SORTCN(1, 5, C3)
      CALL CPR(C3, 15.,C3)
      CALL PICC(C1, C3, N)
      CALL SORTP(2, P1)
      CALL PICC(C2, C3, N)
      CALL SORTP(3, P2)
      CALL PMP(P1, P3, 1)
      CALL PMP(P2, P4, 1)
      CALL PICC(C4, C2, N)
      CALL SORTP(3, P5)
      CALL SORTP(4, P6)
      CALL LPARX(-115., L1)
      CALL CNCTCL(C1, L1, 41.,N)
      CALL SORTP(2, C5)
      CALL CPR(C5, 41.,C5)
      CALL PICC(C1, C5, N)
      CALL SORTP(1, P7)
      CALL PILC(L1, C5, N)
      CALL SORTP(1, P8)
      CALL LPARY(-35.,L2)
      CALL CNCTLL(L1, L2, 10.,N)
      CALL SORTCN(1, 2, C6)
      CALL CPR(C6, 10.,C6)
      CALL PILC(L1, C6, N)
      CALL SORTP(1, P9)
      CALL PILC(L2, C6,N)
      CALL SORTP(1, P10)
      CALL LPARY(31., L3)
      CALL PMLP(L3, P9, P11)
      CALL PMLP(L3, P10, P12)
      CALL CNCTCP(C1, P12,158.,N)
      CALL SORTCN(1, 7, C7)
      CALL CPR(C7, 158.,C7)
      CALL PICC(C7, C1, N)
      CALL SORTP(1, P14)
      CALL CXYR(60.,-70.,20.,C8)
      CALL CNCTCC(C8, C7, 20.,N)
```

```
CALL SORTP(3, C9)
CALL CPR(C9, 20.,C9)
CALL SORTP(4, C10)
CALL CPR(C10, 20.,C10)
CALL PICC(C7, C9, N)
CALL SORTP(1, P15)
CALL PICC(C8, C9, N)
CALL SORTP(1, P16)
CALL PICC(C7, C10, N)
CALL SORTP(1, P17)
CALL PICC(C8, C10, N)
CALL SORTP(1, P18)
```

3.2. Геометрические построения

3.2.1. Запоминание геометрических элементов. Для запоминания геометрических элементов предусмотрен специальный буфер, состоящий из двух позиций, в каждой из которых может быть сохранен один элемент (отрезок прямой, дуга окружности или эллипса). Если программа работает с одним из двух занесенных в буфер элементов, то он всегда берется из текущей позиции буфера. В тех случаях, когда используются оба элемента и когда важен их порядок, будем называть первым элемент в текущей позиции, а вторым - в другой позиции.

Существуют программы записи элементов в буфер и программы считывания элементов. Кроме того, есть подпрограмма STRMOD и функция MODGF. С помощью первой можно управлять адресацией в буфере, т. е. управлять выбором позиций в буфере, к которым производится обращение при записи или считывании элементов. С помощью второй можно опрашивать состояние адресной системы буфера, а также узнавать тип занесенных в буфер элементов.

Наконец, существуют две программы (WRSTR и RDSTR), которые позволяют пересылать информацию между буфером и массивами в программе пользователя как в одну, так и в другую сторону. Таким образом, можно сохранить элемент, записанный в буфере, а впоследствии вновь занести его в буфер не вычисляя.

3.2.2. Программы записи и считывания элементов. Программы записи и считывания элементов разбиты на три группы по типу элемента: отрезок, окружность, эллипс. Каждая программа соответствует одной из рассматривавшихся выше программ рисования (см. табл. 2). Все программы записи имеют имена, начинающиеся с букв WR, а программы считывания - с букв RD. За ними следуют трехбуквенные сокращения имен соответствующих программ рисования.

Программы записи имеют те же параметры, что и соответствующие программы рисования за исключением программ записи отрезка, для которых отсутствует параметр J (перо опущено/ поднято). Это естественно, так как он не относится к геометрическим характеристикам элемента. В программах считывания к параметрам соответствующих программ рисования добавляются координаты начальной точки ($X0$, $Y0$), которые для программ записи и рисования соответствуют текущему положению пера и явно не задаются.

При выполнении программ записи информация об элементе записывается в буфер. При выполнении программ считывания переменным - фактическим параметрам - присваиваются характеристики считываемого геометрического элемента. При этом в пределах каждой группы элементов выбор программы считывания не зависит от того, какой программой был записан элемент. Таким образом, по одним характеристикам элемента можно получить другие.

Если элемент был записан как дуга окружности, можно считывать его как полную окружность, которой эта дуга принадлежит (программой RDCRC), не получая при этом части записанной информации. Можно, наоборот, считывать как дугу окружности элемент, записанный как полная окружность. При этом начальная точка считываемой дуги будет совпадать с конечной, а радиус, проведенный в эту точку, будет иметь нулевой наклон по отношению к оси X.

Таблица 2

	Программы рисования	Программы записи	Программы считывания
Группа прямых	MOVE(X,Y,J) MOVA(DL,TH,J) MOVB(DX,DY,J) MOVC(X,Y,DL,J)	WRMVE(X,Y) WRMVA(DL,TH) WRMVB(DX,DY) WRMVC(X,Y,DL)	RDMVE(X0,Y0,X,Y) RDMVA(X0,Y0,DL,TH) RDMVB(X0,Y0,DX,DY) RDMVC(X0,Y0,X,Y,DL)
Группа окружностей	CIRC(R) ARCIA(R,THO,THF) ARCIB(R,XF,YF,J) ARCIC(XM,YM,XF,YF,J) ARCID(XC,YC,BETA)	WRCRC(R) WRACA(R,THO,THF) WRACB(R,XF,YF,J) WRACC(XM,YM, XF,YF,J) WRACD(XC,YC,BETA)	RDCRC(XC,YC,R) RDACA(X0,Y0,R, THO,THF) RDACB(X0,Y0,R, XF,YF,J) RDACC(X0,Y0,XM,YM, XF,YF,J) RDACD(X0,Y0,XC,YC, BETA)
Группа эллипсов	ELPS(A,B,ALPHA) ARCELA(A,B,ALPHA,THO,THF) ARCELB(A,B,ALPHA,XF,YF)	WRELP(A,B,ALPHA) WRAEA(A,B,ALPHA, THO,THF) WRAEB(A,B,ALPHA, XF,YF)	RDELP(XC,YC,A,B, ALPHA) RDAEA(X0,Y0,A,B, ALPHA,THO,THF) RDAEB(X0,Y0,A,B, ALPHA,XF,YF)

Аналогичная картина имеет место и для эллипсов. При считывании полного эллипса, как дуги, начальная точка будет совпадать с конечной, а радиус, проведенный в эту точку, будет совпадать с полуосью *A*. При считывании отрезка прямой программой RDMVC в качестве дополнительной точки выдается середина записанного отрезка и *DL* > 0 (см. рис. 1.5, *в*). Аналогично, при считывании дуги окружности программой RDACC в качестве дополнительной точки выдается середина записанной дуги и J = 0. Последние правила связаны с тем, что дополнительная точка не имеет непосредственного отношения к элементу и поэтому не запоминается в буфере.

Заметим, что в буфере вместе с характеристиками элемента запоминается его тип (отрезок, окружность, эллипс). Поэтому если делается попытка считать элемент, записанный программой другой группы, то обращение считается неправильным и на печатающее устройство выдается диагностический текст НЕ ТОТ ГЕОМЕТРИЧЕСКИЙ ЭЛЕМЕНТ, затем происходит выход из программы без присвоения значений переменным, указанным в параметрах этой программы.

В заключение подчеркнем, что ни программы записи, ни программы считывания не рисуют изображения и не перемещают пера.

3.2.3. Адресация позиций в буфере и обмен информацией между буфером и массивами пользователя. Выше говорилось, что в буфере есть всего две позиции для запоминания элементов. Хотя эти позиции имеют номера 1 и 2, обращения к ним в командах считывания и записи осуществляется некоторым "безадресным" способом. Для команд записи существует два режима: работа с чередованием позиций и без чередования позиций. При работе с чередованием каждые две последовательные команды записи запоминают элементы в разных позициях. Без чередования все элементы записываются в одну и ту же позицию. Любая команда считывания всегда выбирает элемент из текущей позиции, т.е. из той, в которую происходила последняя запись.

Программа STRMOD(J) устанавливает требуемый режим адресации в буфере и номер текущей позиции. Параметры программы:

J - признак адресации.

С сохранением режима адресации:

J = 0 - меняется номер текущей позиции,

J = 4 - данные первого и второго элементов буфера меняются местами, номер текущей позиции не меняется.

С установкой режима адресации:

$1 \le J \le 3$ - устанавливается режим с чередованием позиций,

$-3 \le J \le -1$ - устанавливается режим без чередования позиций,

$|J| = 1$ - текущей становится позиция 1,

$|J| = 2$ - текущей становится позиция 2,

$|J| = 3$ - номер текущей позиции не меняется.

Первоначально устанавливается режим, соответствующий J = 1.

Функция MODGF(J) позволяет опросить режим адресации, номер текущей позиции, а также узнать, к какому типу принадлежит геометрический элемент, записанный в текущей позиции. Значение единственного входного параметра J определяет вопрос, а значение самой функции - ответ на заданный вопрос:

J = 0 - опрашивается режим адресации и номер текущей позиции:

MODGF(0) > 0 - режим с чередованием позиций,

MODGF(0) < 0 - режим без чередования позиций,

|MODGF(0)| = 1 - позиция 1 - текущая,

|MODGF(0)| = 2 - позиция 2 - текущая;

J = 1 - опрашивается тип геометрического элемента в текущей позиции:

MODGF(1) = 0 - позиция пуста,

MODGF(1) = 1 - записан отрезок прямой линии,

MODGF(1) = 2 - записана окружность или ее дуга,

MODGF(1) = 3 - записан эллипс или его дуга.

Программа WRSTR(STORE) осуществляет перепись буфера (обеих позиций, содержащих 28 вещественных чисел) в выделенный пользователем массив. Имя массива задается параметром STORE.

Программа RDSTR(STORE) выполняет перепись в буфер первых 28 чисел массива, задаваемого параметром STORE.

Эти программы дают возможность временно сохранять элементы, а затем пересылать их в буфер, чтобы вновь оперировать с ними.

Информацию, содержащуюся в буфере, можно использовать при изображении стрелок, размерных линий, а также для выполнения геометрических операций.

Программа ARROW(J) исходит из предположения, что элемент (отрезок прямой или дуга окружности), хранящийся в текущей позиции буфера, нарисован, и "пририсовывает" стрелки на его концах. Входной параметр J имеет следующие значения: J = -1 - стрелка изображается при начальной точке, J = 1 - стрелка изображается при конечной точке, J = 0 - стрелки изображаются при обоих концах.

Стрелка всегда направлена изнутри во вне, а острие совпадает с концевой точкой.

Программа DIMDRO(D, J) изображает размерные линии для основной линии (отрезка прямой или дуги окружности), записанной в буфере. Программа имеет следующие параметры (рис. 3.3):

D - смещение главной размерной линии по отношению к основной линии (в выбранных единицах измерения):

D > 0 - главная размерная линия изображается выше (когда нормаль горизонтальна, то левее) основной линии,

D < 0 - главная размерная линия изображается ниже (когда нормаль горизонтальна, то правее) основной линии,

D = 0 - главная размерная линия совпадает с основной;

J - тип размерных линий:

$|J| < 10$ - расположение стрелок внутреннее,

$|J| \ge 10$ - расположение стрелок наружное,

J < 0 - боковая размерная линия проводится только справа (когда нормаль горизонтальна, то сверху) от нормали,

J > 0 и J ≠ 10 - боковая размерная линия проводится только слева (когда нормаль горизонтальна, то снизу) от нормали,

J = 0 или J = 10 - боковые размерные линии проводятся с обеих сторон от нормали.

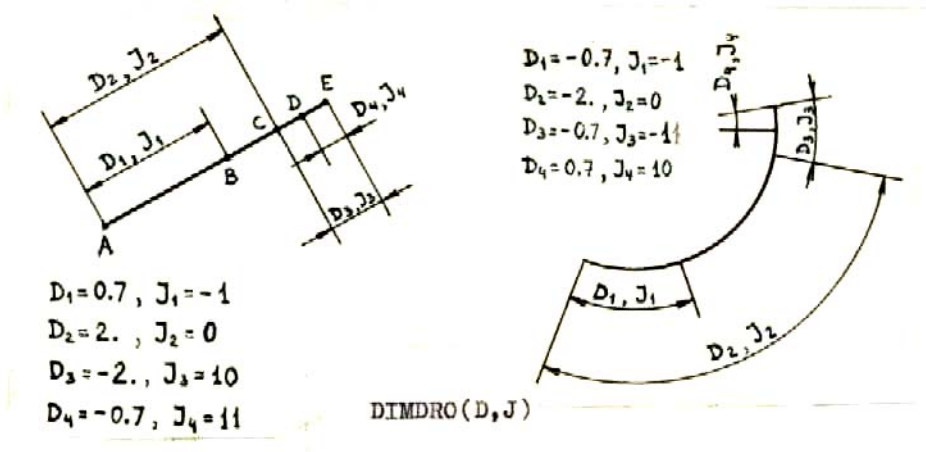

Рис. 3.3. Иллюстрация к программе DIMDRO(D,J).

3.2.4. Операции с геометрическими элементами. Простейшее вычисление состоит в определении координат текущего положения пера и значения выбранной единицы измерения на странице. Дело в том, что перо может попасть в некоторую точку не только при явном задании ее координат. Например, после проведения отрезка заданной длины под заданным углом (MOVA) координаты пера не известны. Точно также нужно считать их неизвестными после того, как нарисован подрисунок по подпрограмме, во внутренние детали которой нет необходимости вникать. В то же время координаты пера могут оказаться нужными для того, чтобы, например, "привязать" к определенному месту поясняющий текст или какие-либо другие элементы изображения.

Возможность опрашивать выбранные единицы измерения оказывается полезной при программировании подрисунков, размеры которых не должны зависеть от единицы измерения. Например, удобно, чтобы надписи, относящиеся к разметке координатных осей при построении графиков, имели постоянный размер. Во всех этих случаях можно воспользоваться программами WHERE или WHERP (см. 1.3).

Функция DIST(J) позволяет определить минимальное расстояние от текущей точки до ближайшей точки, принадлежащей геометрическому элементу, записанному в текущей позиции буфера. При вычислении расстояния имеется возможность "расширения" геометрического элемента: отрезка прямой до полной прямой, дуги окружности до полной окружности, а дуги эллипса до полного эллипса. Параметр J - это признак такого расширения. Если J = 0, то геометрический элемент берется с расширением, а если J = 1, - без расширения. Значение функции DIST и дает искомое расстояние.

Программа DDIST(X, Y) позволяет узнать координаты той точки, расстояние до которой было определено последней выполненной функцией DIST. Между обращениями к функции DIST и программе DDIST могут выполняться любые другие действия кроме обращения к программе CROSS. В этом последнем случае также, как и в случае, когда не было предварительного обращения к функции DIST, программа DDIST выдаст неверные результаты.

Программа SECANT(SL, ALPHA, X, Y) вычисляет координаты конца отрезка заданной длины, начинающегося в текущей точке и проходящего под заданным углом к прямой, которая хранится в текущей позиции буфера (рис. 3.4, *а*). Начальные точки этих прямых обозначены на рисунке соответственно через *X*0, *Y*0 и *X*01, *Y*01. Параметры программы:

SL - длина отрезка (если SL = 0, т. е. не задана длина, концом отрезка считается точка его пересечения с прямой);

ALPHA - угол между отрезком и прямой (в градусах);

X, Y - координаты конца отрезка.

Обращение к программе считается некорректным, если ALPHA = 0° или ALPHA = 180°, а SL = 0, и неправильным, если очередной элемент для считывания не является прямой.

Программа CROSS(X, Y, K) проверяет, пересекаются ли два хранящихся в буфере геометрических элемента и, если пересекаются, то вычисляет координаты точек пересечения. Каждое обращение к программе дает координаты только одной точки. Если геометрические элементы пересекаются в нескольких точках, то для получения всех их необходимо несколько раз обратиться к программе CROSS. Точки пересечения ищутся для "расширенных" геометрических элементов (см. функцию DIST). Программа имеет следующие параметры:

X, Y - координаты очередной точки пересечения;

K - характеристика очередной точки:

K < 0 - выданная точка не последняя,

K > 0 - выданная точка последняя;

K = 0 - элементы не пересекаются, координаты не определены,

|K| = 1 - пересекаются сами элементы,

|K| = 2 - первый элемент пересекает "продолжение" второго,

|K| = 3 - второй элемент пересекает "продолжение" первого,

|K| = 4 - пересекаются "продолжения" элементов.

Если после выдачи последней точки не было записей в буфер и снова происходит обращение к программе CROSS, то значения X, Y, K не определены, на печатающем устройстве выдается диагностический текст ВСЕ ТОЧКИ ВЫДАНЫ. Программа CROSS вычисляет координаты точек пересечения только для прямых и окружностей. Если в какой-либо позиции буфера окажется эллипс, обращение к программе считается неправильным.

Программа LITAN(XT, YT, J) находит точку касания (XT, YT) прямой, проходящей через текущую точку и касающейся окружности в буфере (рис. 3.4, *б*). Параметр J задает вариант касания: J = 1 - касание справа, J = -1 - касание слева. Направления "справа" и "слева" определяются по отношению к лучу, проведенному из текущей точки в центр окружности. Если текущая точка оказывается внутри окружности, обращение к программе считается некорректным. Если в буфере не окружность, обращение к программе считается неправильным.

Программа CIRTAL(R, XT, YT ,J) находит точку (XT, YT), в которой окружность радиуса R, проходящая через текущую точку, касается прямой, записанной в буфере. При J = 1 точка касания ищется справа от луча, проведенного из текущей точки перпендикулярно прямой, при J = -1 - слева (рис. 3.4, *в*). Если расстояние от текущей точки до прямой превышает диаметр окружности, обращение к программе считается некорректным. Если в буфере записана не прямая, то обращение к программе считается неправильным.

Программа CIRTAC(R, XT, YT, J) находит точку (XT, YT), в которой окружность заданного радиуса R касается окружности, хранящейся в текущей позиции буфера (рис. 3.4, г). В дальнейшем будем называть эту окружность основной. Параметр J определяет вариант касания окружностей следующим образом:

|J| = 1 - касание наружное,

|J| = 2 - касание внутреннее,

J > 0 - центр касательной окружности справа от луча, идущего из точки (X0, Y0) в центр основной окружности,

J < 0 - центр касательной окружности слева от того же луча.

Обращение к программе считается некорректным, если:

а) диаметр касательной окружности меньше расстояния от текущей точки до ближайшей точки, лежащей на основной окружности,

б) при внутреннем касании диаметр касательной окружности меньше (текущая точка вне окружности) или больше (текущая точка внутри окружности) расстояния от текущей точки до самой удаленной точки основной окружности;

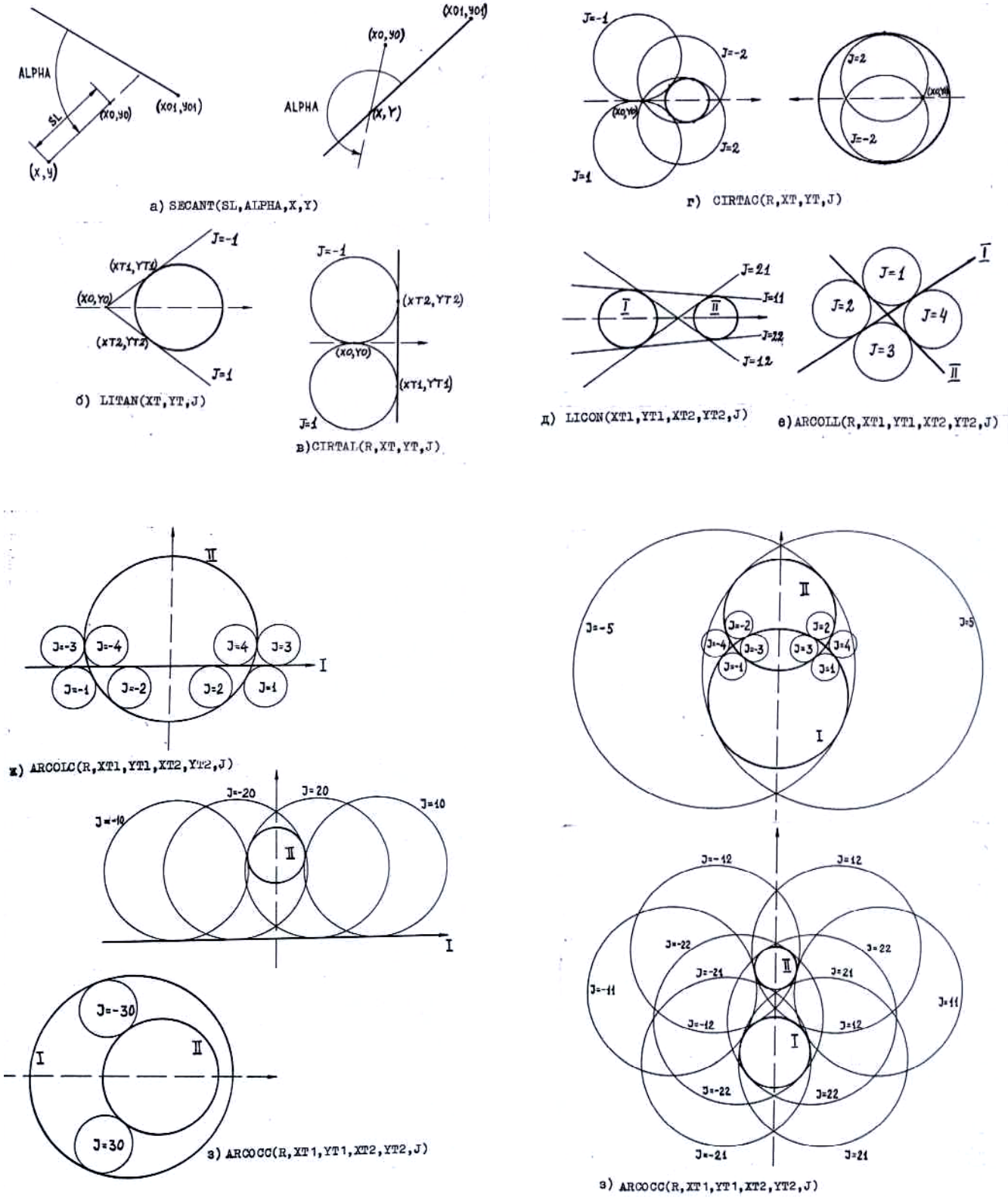

Рис. 3.4. Иллюстрации к программам геометрических построений.

в) касание задано наружное, а текущая точка лежит внутри основной окружности;

г) текущая точка совпадает с центром основной окружности.

Если при обращении к программе CIRTAC окажется, что в текущей позиции буфера записана не окружность, то обращение к программе считается неправильным.

Программа LICON(XT1, YT1, XT2, YT2, J) находит точки касания общей касательной прямой для двух окружностей, записанных в буфере. Программа имеет следующие параметры (см. рис. 3.4, *д*):

XT1, YT1, XT2, YT2 - координаты точек касания на первой и второй окружностях;

J - вариант касания:

J = 11 - обе точки касания слева от линии центров,

J = 12 - точка касания первой окружности находится слева от линии центров, а второй - справа,

J = 21 - точка касания первой окружности находится справа от линии центров, а второй - слева,

J = 22 - обе точки касания справа от линии центров.

Линия центров в этом случае направлена от центра первой окружности к центру второй окружности.

Обращение к программе считается некорректным, если одна окружность находится внутри другой, или окружности пересекаются, а точки касания требуется найти по разные стороны от линии центров. Если же в буфере хотя бы один элемент не является окружностью, то обращение к программе считается неправильным.

Программа ARCOLL(R, XT1, YT1, XT2, YT2, J) находит точки касания общей касательной окружности для двух пересекающихся прямых, записанных в буфере. Параметры программы (рис. 3.4, *е*):

R - радиус касательной окружности;

XT1, YT1, XT2, YT2 - координаты точек касания на первой и второй прямых;

J - номер квадранта, образованного прямыми, в котором должна находиться касательная окружность,

Квадранты нумеруются, начиная с единицы, против часовой стрелки от положительного направления первой прямой. Направление второй прямой несущественно. Если прямые параллельны, то обращение к программе считается некорректным. Если хотя бы в одной позиции буфера записана не прямая, то обращение к программе считается неправильным.

Программа ARCOLC(R, XT1, YT1, XT2, YT2, J) находит точки касания общей касательной окружности для прямой и окружности, записанных в буфере. Параметры программы (рис. 3.4, *ж*):

R - радиус общей касательной окружности;

XT1,YT1, XT2, YT2 - координаты точек касания на первом и втором элементах в буфере;

J - вариант касания:

J > 0 - центр касательной окружности находится справа,

J < 0 - центр касательной окружности находится слева,

|J| < 10 - прямая и окружность пересекаются, образуя два сегмента: |J| = 1 - наружное касание со стороны меньшего сегмента, |J| = 2 - внутреннее касание со стороны меньшего сегмента, |J| = 3 - наружное касание со стороны большего сегмента, |J| = 4 - внутреннее касание со стороны большего сегмента.

|J| ≥ 10 - прямая и окружность не пересекаются: |J| = 10 - наружное касание, |J| = 20 - внутреннее касание.

Расположение "слева" или "справа" для центра касательной окружности определяется относительно перпендикуляра к записанной в буфере прямой, проходящего через центр записанной в буфере окружности. Перпендикуляр направлен от прямой к центру окружности, если прямая находится в текущей позиции буфера, и наоборот, от центра окружности к прямой, если в текущей позиции находится окружность.

Когда прямая проходит через центр окружности, большим условно считается сегмент, находящийся слева от самой прямой с учетом ее направления от начала к концу.

Обращение к программе считается некорректным, если:

а) |J| < 10, а занесенные в буфер окружность и прямая не пересекаются, либо, наоборот, |J| ≥ 10, а они пересекаются;

б) |J| < 10 и четно (внутреннее касание), а окружность радиуса R не может поместиться внутри заданного сегмента,

в) |J| ≥ 10, и диаметр касающейся окружности меньше при наружном касании минимального, а при внутреннем касании максимального расстояния от прямой до точки, принадлежащей окружности из буфера.

Если в буфер занесены не прямая и окружность, то обращение к программе считается неправильным.

Программа ARCOCC(R, XT1, YT1, XT2, YT2, J) находит точки касания общей касательной окружности для двух окружностей из буфера. Программа имеет следующие параметры (рис. 3.4, *з*):

R - радиус общей касательной окружности;

XT1,YT1, XT2,YT2 - координаты точек касания соответственно на первой и второй окружностях в буфере;

J - вариант касания:

J > 0 - центр касательной окружности справа от линии центров,

J < 0 - центр касательной окружности слева от линии центров (пусть I обозначает первую, а II - вторую окружность в буфере),

|J| < 10 - окружности, записанные в буфере, пересекаются:
|J| = 1 - касание с I внутреннее, а с II - наружное, |J| = 2 - касание с I наружное, а с II - внутреннее (касательная окружность внутренняя по отношению к II), |J|= 3 - касание с I и II внутреннее (касательная окружность внутренняя по отношению к I и II), |J| = 4 - касание с I и II наружное, |J| = 5 - касание с I и II внутреннее (касательная окружность внешняя по отношению к I и II).

|J| > 10 - окружности, записанные в буфере, не пересекаются:
|J| = 11 - касание с I и II наружное, |J| = 12 - касание с I наружное, а с II - внутреннее (касательная окружность внешняя по отношению к II), |J| = 21 - касание с II наружное, а с I - внутреннее (касательная окружность внешняя по отношению к I), |J| = 22 - касание с I и II внутреннее (касательная окружность внешняя по отношению к I и II), |J| = 30 - с меньшей окружностью касание наружное, а с большей - внутреннее.

Линия центров здесь считается направленной от первого элемента ко второму.

Обращение к программе ARCOCC считается некорректным, если:

а) |J| < 10, однако занесенные в буфер окружности не пересекаются, либо наоборот, |J| > 10, а окружности пересекаются;

б) диаметр внутренней окружности недостаточно мал, либо диаметр наружной окружности недостаточно велик в многочисленных случаях внутреннего касания;

в) окружности не пересекаются и удалены друг от друга более, чем на диаметр касающейся окружности;

г) окружности находятся одна внутри другой, а |J| ≠ 30.

Если в буфер занесена не пара окружностей, обращение считается неправильным.

При реализации многих программ, описанных в этом параграфе, требовалось определить угол наклона отрезка к оси или же выяснить, где находится значение некоторой величины по отношению к окрестности некоторой точки на числовой оси. Для этих целей предназначены подпрограммы-функции ANGLER и IVEST.

Функция ANGLER(DELTAX, DELTAY) для отрезка прямой, заданного приращениями координат DELTAX и DELTAY, вычисляет угол его наклона к оси *X*. Значение функции дает величину угла в градусах.

Функция IVEST(A, B, EPS) определяет, где находится значение заданной переменной по отношению к окрестности некоторой точки на числовой оси. Параметры программы:

A - проверяемая величина;

B - центр окрестности на числовой оси;

EPS - радиус окрестности.

Значение самой функции отвечает на заданный вопрос следующим образом:

IVEST = -1 - значение лежит левее окрестности (A<(B-EPS)),

IVEST = 0 - значение принадлежит окрестности ((B-EPS) ≤ A ≤ (B+EPS)),

IVEST = 1 - значение лежит правее окрестности ((B+EPS)<A).

3.2.5. Примеры. В качестве примера использования описанных выше программных средств рассмотрим чертеж ступицы, изображенный на рис. 3.5, а также два фрагмента программы построения этого чертежа:

а) Фрагмент программы, формирующий внешний контур чертежа ступицы с N зубьями:

```
CALL WHERE(XC, YC, F)
CALL WRCRC(D(5) / 2)
CALL MOVB(D(5) / 2, 0, 0)
CALL WRCRC(R)
CALL CROSS(X1,Y1, J)
CALL CROSS(X2,Y2, J)
TH = ANGLER(X1 - XC,Y1 - YC)*0.01745329252
CALL MOVE(X2,Y2, 0)
DTH = 360./N*0.01745329252
TH1 = TH
TH2 = DTH - TH
DO 1 I = 1, N
X1 = XC + D(5) / 2*COS(TH1)
```

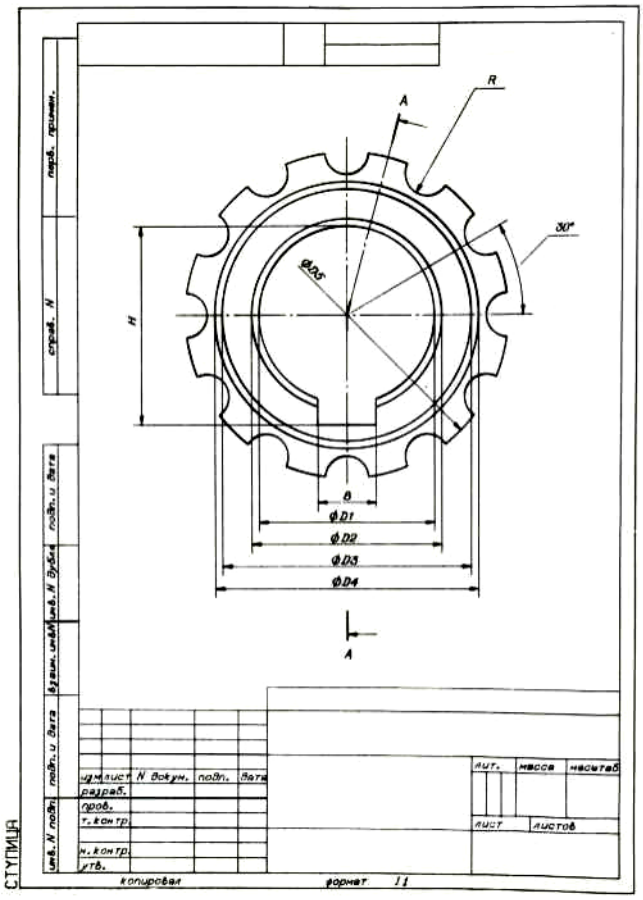

Рис. 3.5. Пример построения чертежа.

```
X2 = XC + D(5) / 2*COS(TH2)
Y1 = YC + D(5) / 2*SIN(TH1)
Y2 = YC + D(5) / 2*SIN(TH2)
CALL FATARC(-R, X1,Y1, 0, 0.5)
```

```
   CALL FATARC(D(5) / 2, X2,Y2, 0, 0.5)
   TH1 = TH1 + DTH
   TH2 = TH2 + DTH
1 CONTINUE
```

б) Фрагмент программы, формирующий основные размерные линии:

```
   CALL MOVE(XC - B/2,Y1, 0)
   CALL WRMVE(XC + B/2, Y1)
   CALL DIMDRO(-3.5, 0)
   DO 2 J = 1, 4
   CALL MOVE(XC - D(J) / 2,YC, 0)
   CALL WRMVE(XC + D(J) / 2,YC)
   CALL DIMDRO(-6.25-0.75 * J, 0)
2 CONTINUE
```