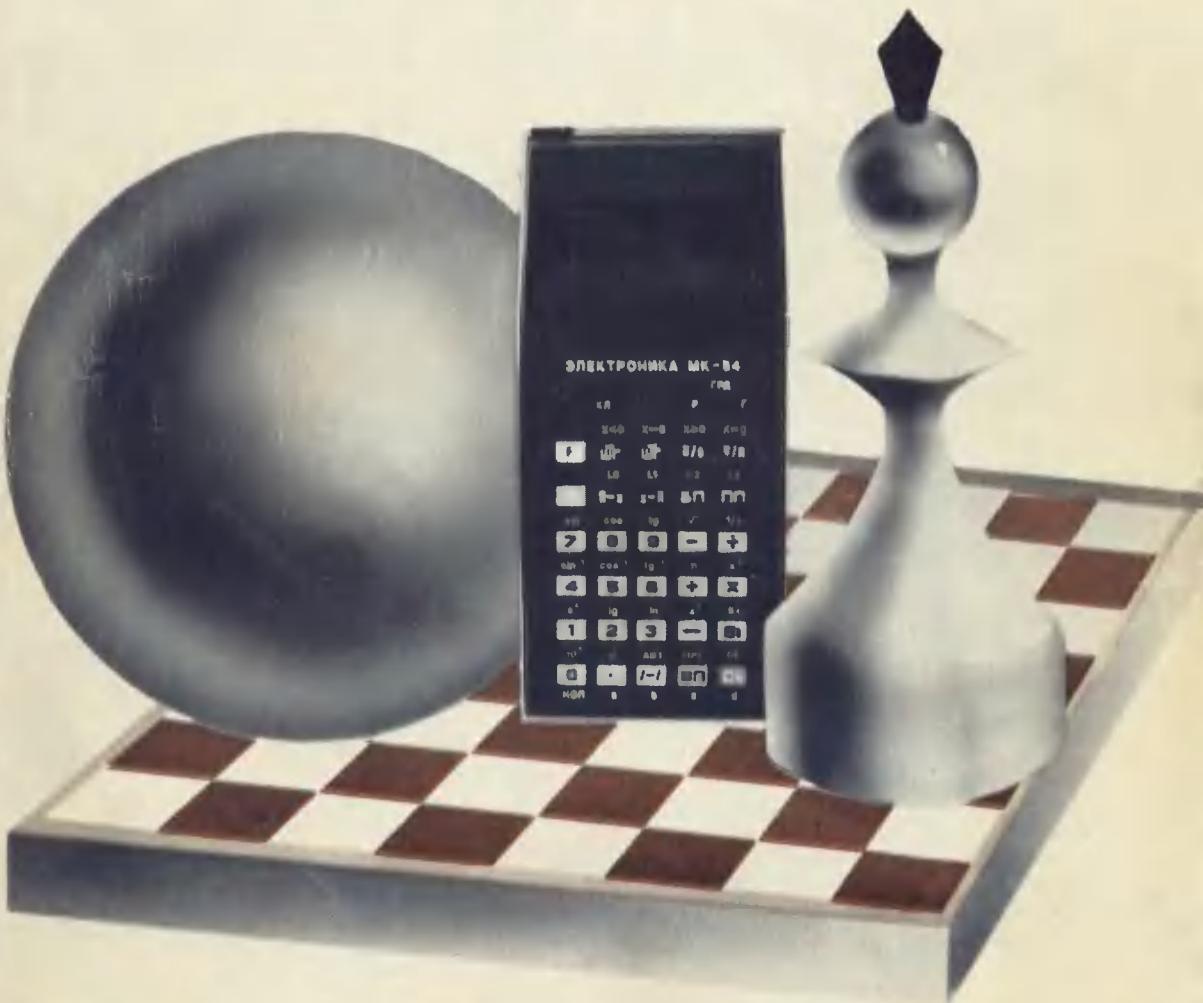


Я.К. ТРОХИМЕНКО

ИГРЫ
с МИКРО-
ЭВМ



Я.К.ТРОХИМЕНКО

ИГРЫ
с МИКРО-

ЭВМ

Киев
«Техніка»
1986

32.97

Т76

УДК 621.322:621.382.049.77

Трохименко Я. К.

Т76 Игры с микро-ЭВМ.— К.: Техніка, 1986.— 120 с., ил.— Библиогр.: с. 119.
55 к. 35 000 экз.

Как связаны игры с кибернетикой и проблемой искусственного интеллекта? Как использовать микро-ЭВМ в качестве партнера или судьи в игре? Ответы на эти вопросы содержатся в данной научно-популярной книге, где рассматривается составление алгоритмов разнообразных занимательных игр и описывается 60 игровых программ для наиболее массовых универсальных микро-ЭВМ — программируемых микрокалькуляторов. Предназначена для широкого круга читателей.

Т 2405000000-166 54.86 32.97
М202(04)-86

Рецензенты канд. физ.-мат. наук

Г. В. Поддубный, Ю. М. Польский

Редакция литературы по энергетике,
электронике, кибернетике и связи

Зав. редакцией З. В. Божко

ПРЕДИСЛОВИЕ

Развитие вычислительной техники в настоящее время характеризуется качественно новым этапом, связанным с массовым внедрением универсальных микро-ЭВМ (персональных ЭВМ и программируемых микрокалькуляторов), отличающихся малой стоимостью машинного времени, простотой эксплуатации и доступностью широкому кругу пользователей. Это создает предпосылки для существенного повышения производительности труда практически во всех отраслях народного хозяйства, и поэтому массовое освоение методики решения различных задач с помощью ЭВМ становится столь же необходимым, как в прошлом борьба с безграмотностью.

Методике решения вычислительных задач с помощью универсальных ЭВМ, в основном применимой и для персональных микро-ЭВМ, посвящена обширная литература, а в последние годы издано достаточно много книг (например, работы [8, 9, 11, 12]) по вычислениям на программируемых микрокалькуляторах. Значительно меньше внимания уделено в литературе решению с помощью ЭВМ логических задач, связанных с определением требуемых связей (структуры) между элементами заданных систем. Навыки в решении подобных задач проще всего (в особенности для пользователей, работа которых не связана с вычислениями) приобрести на примерах решения занимательных игровых задач. Кстати, быстрый рост производства домашних ЭВМ во всем мире в основном обусловлен их применением для различных игр. В отечественной литературе эта тема затронута лишь применительно к программируемым микрокалькуляторам в книге [7] и, частично, в книге [10], где приведен ряд игровых программ для первого отечественного программируемого микрокалькулятора «Электроника Б3-21».

В настоящей книге рассмотрена методика составления алгоритмов разнообразных занимательных логических и игровых программ для программирования универсальных микро-ЭВМ различных классов. Для большинства персональных ЭВМ используются различные версии языка программирования БЕЙСИК, основная версия которого описана в книге. Однако для этих микро-ЭВМ используются и различные другие языки программирования, также значительно отличающиеся от входных языков наиболее массовых универсальных микро-ЭВМ — программируемых микрокалькуляторов. Поэтому рассматриваемые алгоритмы решения логических и игровых задач иллюстрируются

в этой книге программами на входном языке программируемых микрокалькуляторов «Электроника Б3-34», «Электроника Б3-54», «Электроника МК-54», «Электроника МК-56», «Электроника МК-61» и «Электроника МК-52».

Содержащиеся в книге сведения можно использовать различным образом. Будущие пользователи микро-ЭВМ найдут в ней общие сведения о языках программирования и методах составления алгоритмов решения игровых задач. Пользователи программируемых микрокалькуляторов смогут применить приведенные программы для проведения занимательного и полезного досуга и приобрести навыки, необходимые для решения многих прикладных задач. Пользователи персональных ЭВМ смогут использовать рассмотренные алгоритмы для программирования своих ЭВМ.

Программирование — не простая задача, но эта книга рассчитана на широкий круг читателей, так как в ней даны определения всех используемых понятий, выходящих за пределы основного объема знаний, полученных в средней школе.

Отзывы и пожелания просьба направлять по адресу: 252601, Киев, 1, Крещатик, 5, издательство «Техника».

Глава

1

ИГРЫ И КИБЕРНЕТИКА

В нашем сознании слово *игра* чаще всего ассоциируется с игрушками, детскими забавами или чем-то еще не очень серьезным. Между тем игры имеют первостепенное значение как в жизни отдельных людей, так и в развитии человеческого общества. Согласно строгому определению, игрой называется моделирование конфликтных ситуаций между двумя или несколькими сторонами, по крайней мере, одна из которых преследует определенную цель. Сторонами в конфликте могут быть люди, но обычно человек достигает своих целей в противоборстве с силами природы. Моделирование выхода из таких конфликтных ситуаций называют играми с природой.

Поведение простейших животных в их относительно несложных жизненных ситуациях заложено в генетическом коде, отображающем опыт множества поколений, накопленный в процессе естественного отбора. Даже цыпленок, только что вылупившийся из яйца, начинает клевать зерна подобно взрослой курице, так как такое поведение обусловлено его генетическим кодом. Однако чем выше уровень биологической организации животного, тем больше его поведение зависит от рефлекторных навыков, приобретенных в течение жизни. Сложная система таких навыков, определяющих поведение в конфликтных ситуациях, вырабатывается у высших животных преимущественно в играх. Поэтому склонность котят или волчат к играм объясняется не

их шаловливостью, а жесткой необходимостью приобретения навыков для выживания без помощи родителей.

В жизни человека игры имеют еще большее значение. Малыши, тянувшиеся к погремушкам или складывающие кубики, приобретают в играх рефлекторные навыки координации движений и познают на собственном опыте свойства окружающего мира. Непрерывной игрой является и обучение в школе, так как классные сочинения или арифметические упражнения лишь моделируют возможные жизненные ситуации. Человек приобретает и закрепляет в памяти необходимые навыки и после школы в различных тренировках, спортивных, деловых, военных и прочих играх. Даже решение кроссвордов или игра в домино служат развитию сообразительности и навыков, требуемых для решения разнообразных жизненных задач.

Многие навыки поведения, закрепляемые в подсознании человека, обеспечивают автоматизм движений, необходимый не только для ходьбы или игры в теннис, но и для выполнения большинства производственных операций. А закрепленные в памяти знания о возможных подходах к решению различных задач образуют жизненный опыт человека и определяют то, что называется здравым смыслом. Вместе с этим человеку часто приходится встречаться с новыми задачами, способ решения которых ему неизвестен. Подобная задача мо-

жет возникнуть неожиданно и требовать немедленного решения. В таких случаях человек действует рефлекторно, но при отсутствии нужных навыков принятное решение может привести к нежелательным или даже трагическим результатам. Если же человек располагает временем для поиска выхода из конфликтной ситуации, какой всегда является новая задача, то он мысленно или используя бумагу в качестве «внешней памяти» моделирует возможные пути решения задачи и выбирает лучшие из них, т. е. занимается именно тем, что по определению называется игрой. Поэтому известные слова «Что наша жизнь? Игра!» из «Пиковой дамы» А. С. Пушкина уместно рассматривать не как броскую фразу литературного героя, а как глубокое обобщение гениальным поэтом существа интеллектуальной деятельности человека.

Проигрывание человеком поиска наилучшего выхода из конфликтной ситуации в общем случае можно разбить на несколько взаимосвязанных этапов. Человек прежде всего формализует условия задачи, отбрасывая все несущественное и сохраняя лишь те исходные данные и связи между ними, которые позволяют найти искомый результат. Затем человек выбирает или даже изобретает метод (общий подход) решения подобных задач, помогающий найти способ решения рассматриваемой задачи. Лишь после такого мысленного или теоретического моделирования он может приступить к действиям, необходимым для решения задачи.

Методы решения практических любых задач можно разбить на две группы. К первой группе относятся прямые методы с заранее известным числом операций (действий), необходимых для получения требуемого результата, ко второй — косвенные ме-

тоды последовательных приближений, при использовании которых число операций заранее неизвестно и их прекращают лишь после получения требуемого результата.

Способ решения конкретной задачи отображают алгоритмом — конечной последовательностью однозначных описаний операций, выполнимых исполнителем алгоритма и приводящих к искомому результату. Если исполнитель не может (например, не умеет или не располагает необходимыми средствами) выполнить операцию, то ее описание в алгоритме должно быть заменено описаниями выполнимых операций. Например, при заданных числах a , b , c и d большинство школьников сумеют найти число y по формуле $y = a - b \ln(c/d)$, которая и служит для них алгоритмом решения задачи. Однако для тех, кто забыл правила записи формул, необходимо указать, что вначале вычисляют частное c/d , находят его логарифм, произведение которого на число b вычитают из числа a .

Алгоритмы представляют словесно-формульными описаниями, схемами или программами. Словесно-формульное описание состоит из перечня описаний операций, обычно обозначаемых порядковыми номерами и называемых шагами алгоритма. Описания операций разделяют на два основных типа.

Описания первого типа, соответствующие операциям с однозначно определенным результатом, называют операторами * присваивания. Формально их можно представить предложениями вида «пусть $I = A$ », где слева

* Оператор — тот, кто (или то, что) управляет выполнением операций. В математике оператором называют правило, обозначенное некоторым символом, которым может быть словесное описание, и ставящее в соответствие элементы двух множеств (например, множеств исходных данных и результатов вычислений).

от знака равенства записан символ (*идентификатор*) объекта, а справа — символ состояния, которое присваивается этому объекту после выполнения операции. Например, описание операции «включить свет» является логическим оператором присваивания, так как его можно представить предписанием «пусть $C = 1$ », где символом C обозначен свет, а символам 1 и 0 соответствуют его включение и выключение. При вычислениях в арифметических операторах присваивания слева от знака равенства записывают символ (*имя*) переменной, а справа — число или выражение, результат вычисления которого в дальнейшем принимается равным переменной. Так, оператор «пусть $x = x + 1$ » означает, что в дальнейшем переменной x присваивается значение, на единицу большее предыдущего.

Описания второго типа, называемые условными операторами или операторами условных переходов, соответствуют операциям, результат выполнения которых зависит от определенного условия. Обычно их представляют предложениями вида «если выполнено условие Y , то перейти к шагу m , иначе — к шагу n ».

Заметим, что в таких наиболее распространенных сборниках алгоритмов, как поваренные книги чаще всего встречаются операторы присваивания («взять морковь, посыпать солью...»), но иногда там можно найти и условные операторы, например «если маринад посветлеет, то прекратить кипячение, иначе добавить уксус...».

Для наглядности алгоритмы представляют схемами, в которых операторы присваивания обозначены прямоугольниками, а условные операторы — ромбами с выходами «Да» и «Нет», соответствующими выполнению и невыполнению проверяемого

условия. Все блоки схемы соединяют линиями со стрелками, указывающими последовательность выполнения операций, а начало и конец алгоритма обозначают овалами.

В качестве простейшего примера рассмотрим алгоритм кипячения воды, налитой в стоящий на плите чайник. Для большинства людей такой алгоритм достаточно описать оператором присваивания «вскипятить воду», но для очень рассеянного и занятого человека придется составить, например, следующее описание:

0. Включить плиту.
1. Зайти на кухню через 20 мин.
2. Выключить плиту.

Этот простейший алгоритм (рис. 1) формально невыполним, если его исполнитель располагает лишь персональными часами с « заводом » на 5 мин. В этом случае придется расширить описание, например:

0. Включить плиту.
1. Перевернуть часы и дождаться высыпания песка.
2. Перевернуть часы и дождаться высыпания песка.
3. Перевернуть часы и дождаться высыпания песка.



Рис. 1.

4. Перевернуть часы и дождаться высыпания песка.
5. Зайти на кухню и выключить плиту.

Алгоритмы, реализующие методы последовательных приближений, содержат описания операций, многократно повторяемых до получения искомого результата. Такие замкнутые последовательности операций называют циклами, а их однократное выполнение — итерацией. Итерационные циклы могут содержаться и в алгоритмах, реализующих прямые методы, но в этом случае число итераций заранее задается. В частности,

последнее из описаний алгоритма кипячения воды можно сократить, организовав итерационный цикл (рис. 2):

0. Включить плиту.

1. Перевернуть часы и дождаться высыпания песка.

2. Если часы переворачивались четыре раза, то перейти к шагу 3, иначе перейти к шагу 1.

3. Зайти на кухню и выключить плиту.

Рассмотренные алгоритмы, в которых реализован прямой метод решения задачи, отличаются тем недостатком, что искомый результат (закипевшая вода) зависит от интенсивности горения плиты, и через 20 мин вода может выкипеть или еще не закипит. В данном случае более эффективен метод последовательных приближений, представляемый следующим описанием алгоритма для исполнителя с обычными часами:

0. Включить плиту.

1. Зайти на кухню через 10 мин.

2. Зайти на кухню через 5 мин.

3. Если вода кипит, то перейти к шагу 4, иначе — к шагу 2.

4. Выключить плиту.

Этот алгоритм (рис. 3) обеспечивает решение задачи в широком интервале интенсивности горения плиты.

Словесно-формульное описание (например, инструкция по использованию огнетушителя) или схема алгоритма являются для исполнителя и программой действиями, но в более узком смысле программой называют представление алгоритма в виде последовательности символов, отображающих необходимые операции и порядок их выполнения. Например, неразветвленный алгоритм, представленный схемой на рис. 1, можно отобразить программой ВП К20 СТОП, где каждый шаг алгоритма обозначен определенным символом, характеризующим выполняемую операцию.



Рис. 2.

Некоторые трудности возникают при отображении «одномерной» программой разветвленного алгоритма с переходами к шагам, следующим не по порядку. Эти трудности преодолевают двумя основными способами. Первый способ заключается в нумерации каждого шага программы адресом (кодом порядкового номера), например 00, 01, 02,..., для программ, содержащих менее 100 шагов. В этом случае условные операторы удобно описывать предложением вида «если выполнено условие У, то перейти к следующему шагу, иначе перейти к шагу с адресом А». В программах такие условные операторы удобно отображать символами вида У?А, где У — проверяемое условие, А — адрес перехода при его невыполнении. Обязательные (безусловные) переходы отображают в программах операторами или командами безусловного перехода вида БП А. Операторы условного и безусловного перехода обычно занимают по два шага программы, причем второй шаг занимает адрес А этих переходов.

При использовании адресов (00, 01, 02, ...) шагов программы алгоритм, схема которого показана на рис. 3, можно представить программой

БП К10 К5 ВК? 02 СТОП, согласно которой при выполнении проверяемого условия («вода кипит?») следует перейти к заключительной операции («выключить плиту»), а при невыполнении перейти к шагу с адресом 02 («зайти на кухню через 5 мин»).

Второй способ заключается в использовании специальных символов, называемых метками и записываемых в программе после команды проверки условия У? или команды безусловного перехода БП и перед тем оператором программы, к выполнению которого надлежит перейти. Например, выбрав для метки символ М1, схему

алгоритма, показанную на рис. 3, можно представить программой
ВП К10 М1 К5 ВК? М1 СТОП.

В этом случае при невыполнении проверяемого условия (ВК?) исполнитель алгоритма должен перейти к оператору К5, перед которым в программе записана такая же метка М1.

Очевидно, что для понимания алгоритма, представленного программой, необходим толковый словарь с перечнем всех используемых символов (лексики) и объяснением их смысла (семантики), а также знание правил записи этих символов (орфографии)

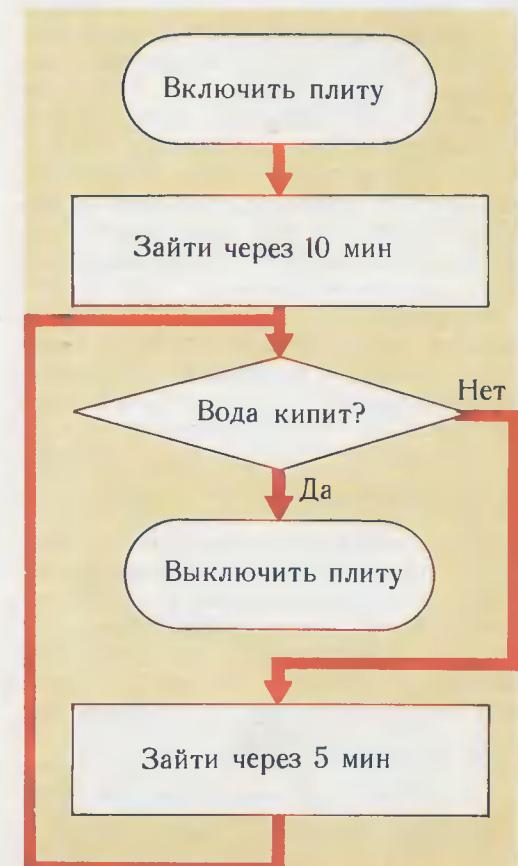


Рис. 3.

и их последовательностей (синтаксиса). Другими словами, для чтения алгоритма необходимо знать язык программирования, на котором составлена программа.

Исполнителем алгоритма может быть не только человек, но и автомат, конструкция которого обеспечивает выполнение всех действий, определяемых операторами программы. Подобные автоматы известны с глубокой древности и выполняют как операции присваивания, так и операции выбора следующего шага в зависимости от выполнения проверяемого условия. Устройства для хранения информации о программе действий в таких автоматах обычно непосредственно связаны с исполнительными механизмами. Примером автоматов с неразветвленными программами могут служить музыкальные шкатулки, в которых операторы присваивания реализованы отверстиями во вращающихся дисках или выступами на вращающихся валиках, являющимися одновременно и элементами исполнительных механизмов для создания звуков различного тона. Более современные кассовые автоматы метрополитена реализуют условный оператор «если пассажир бросил пять копеек, то оставить дверь открытой, иначе закрыть дверь».

Различные механические и электромеханические автоматы с фиксированными программами широко применяют и сегодня, причем к автоматам относятся все машины (например, паровые), действующие без непосредственного участия человека. Однако подлинную революцию в науке и технике вызвало создание в 40-х гг. нашего столетия электронных автоматов, называемых компьютерами (вычислителями) или цифровыми электронными вычислительными машинами (ЭВМ). Их основные особенности заключаются в большой скорости

выполнения операций и, главное, возможности быстрой смены программы и, следовательно, решаемой задачи.

Для хранения и передачи информации внутри ЭВМ обычно используют бинарные (двоичные) коды, содержащие только символы 0 и 1. Такие коды отличаются высокой надежностью, так как их символы сопоставимы с легко различимыми состояниями «Есть» и «Нет» материальных объектов. Каждой бинарной позиции, содержащей символ 0 или 1, соответствует минимальное количество информации, называемое битом.

Числа в бинарных кодах чаще всего представляют в двоичной системе счисления последовательностью $a_r \dots a_1 a_0$ бинарных позиций, называемых двоичными разрядами. Символ 1 в разряде a_0 отображает единицу, а в каждом следующем разряде — число, в два раза большее числа, отображаемого тем же символом 1 в соседнем правом разряде. Сумма чисел, отображаемых единицами во всех $r+1$ разрядах, равна десятичному представлению числа. Например, двоичное представление числа $a_2 = 10101$ соответствует десятичному представлению $a_{10} = 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 21$.

Иногда для представления чисел используют и другие бинарные коды, например, десятично-двоичный, в котором каждый десятичный разряд отображается четырьмя двоичными. Так десятичное представление числа 21 отображается десятично-двоичным представлением 00100001. Добавим, что каждые четыре разряда двоичного представления числа соответствуют одному разряду шестнадцатеричного представления числа, цифры которого обычно обозначают символами 0, 1, ..., 9, A, B, C, D, E, F. Поэтому десятичное представление $a_{10} = 21 = 1 \cdot 16^1 + 5 \cdot 16^0$ отобража-

ется шестнадцатеричным представлением $a_{16} = 15$ или двоичным $a_2 = 00010101$.

Двоичное n -битовое представление информации можно регистрировать (хранить), например, с помощью регистра-ящика с n отделениями, соответствующими бинарным позициям, и камешками, заменяющими символ 1. Передавать эту информацию можно с помощью переносного лотка, в котором камешки размещены соответственно их положению в исходном регистре-ящике.

В ЭВМ *регистры* образованы последовательностями из n электронных элементов, принимающих под воздействием управляющих электрических сигналов одно из двух состояний («включено» и «выключено»), соответствующих символам 1 и 0. Роль лотка для пересылки информации выполняют электрические сигналы длительностью n равных отрезков времени (тактов) с включением напряжения в такты, соответствующие битам с символом 1, и включением напряжения в такты, соответствующие битам с символом 0. Такие сигналы, передаваемые или хранимые в памяти ЭВМ, характеризуются форматом,

определяющим общее число тактов (бит) и их распределение для представления чисел, включая знак и положение запятой, команд управления и адресов тех узлов машины, которым передается информация.

Основные узлы ЭВМ (рис. 4) — *процессор П*; *оперативное запоминающее устройство ОЗУ*, или *память* для временного хранения данных и программы решения задачи; *накопители информации НИ* для ее длительного хранения; *управляющее устройство УУ* для обеспечения взаимодействия всех узлов машины и *устройства ВВ* для *ввода* и *вывода* информации.

Процессор содержит несколько регистров для хранения *операндов* (чисел, над которыми выполняются операции) и промежуточных результатов, а также арифметико-логическое устройство для выполнения арифметических и логических операций над операндами по командам программы.

Программы работы ЭВМ первого (лампового) поколения составляли непосредственно на языках *машинных* (бинарных) кодов в виде длинных последовательностей нулей и единиц.

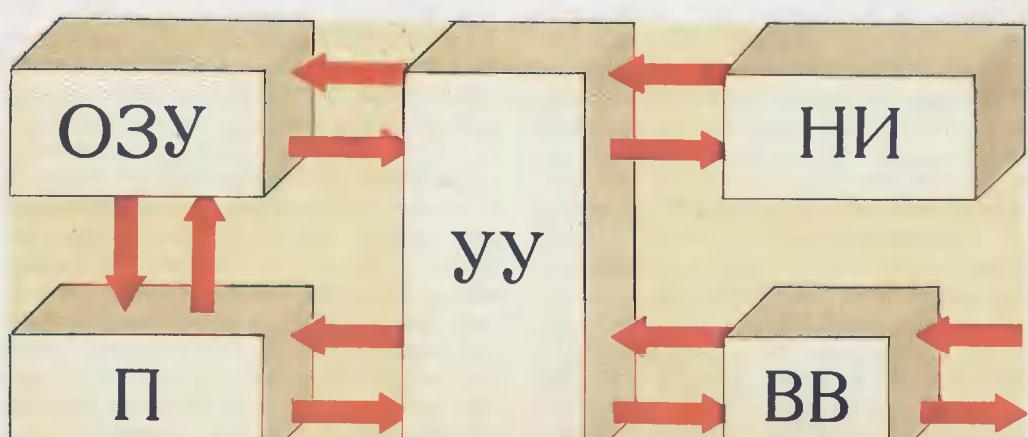


Рис. 4.

Составление таких программ и их ввод в память ЭВМ для каждой задачи чрезвычайно утомительны. Поэтому вскоре для ввода программ стали использовать *ассемблеры* — устройства, формирующие машинные коды команд при нажатии клавиш с легко запоминаемыми сочетаниями букв и цифр, обозначающих элементарные команды. Программирование на языках ассемблеров, называемых также мнемокодами и автокодами, особенно удобно для специализированных (например, управляющих производственными процессами) ЭВМ, программы работы которых изменяются сравнительно редко.

Однако даже простейшие операторы алгоритмов решения задач отображаются большим числом команд на языках ассемблеров. Поэтому еще в 50-х гг. для универсальных ЭВМ, предназначенных для решения разнообразных математических задач, были разработаны *алгоритмические языки* программирования, слова которых непосредственно отображают операторы алгоритмов. Для этого еще машины второго поколения (на полупроводниковых элементах) стали снабжать *трансляторами* — устройствами, обеспечивающими отображение каждого слова входного алгоритмического языка при его вводе в машину соответствующей последовательностью команд в машинных кодах, называемой микропрограммой. Внедрение алгоритмических языков и трансляторов существенно упростило программирование и обеспечило возможность использования ЭВМ широким кругом специалистов.

Любые вычисления в ЭВМ сводятся к последовательности арифметических операций, которые реализуются в процессоре с помощью простейших логических операций (подобных «0 пишем, 1 в уме»). Поэтому, хотя большинство языков программи-

рования предназначено для вычислений, с помощью ЭВМ оказалось возможным решать логические задачи. Эта возможность, ранее связываемая лишь с интеллектуальными способностями человека, обусловила постановку основной проблемы кибернетики — создание искусственного интеллекта в виде машины, способной находить пути решения новых для нее задач.

Первые успехи в программировании ЭВМ для решения шахматных этюдов, перевода иностранных текстов, сочинения музыкальных произведений и стихов породили в 50-х гг. бурную дискуссию на тему «Может ли мыслить ЭВМ?». Было высказано множество категорических мнений, например, о том, что через десятилетие машина станет чемпионом мира по шахматам, переводчикам уготовлена участь динозавров, а поэты переводчики квалифицируются в литературных критиков, завистливо оценивающих успехи машин. Некоторые специалисты утверждали, что для создания искусственного интеллекта достаточно построить машину с памятью, не уступающей человеческой по емкости запоминаемой информации.

С тех пор прошло три десятилетия. Созданы ЭВМ третьего (на интегральных элементах) и четвертого (на элементах с высокой степенью интеграции) поколений. Существуют вычислительные машины размером с наперсток, превосходящие по ряду показателей первые ЭВМ, занимавшие отдельные здания. Созданы супер-ЭВМ с фантастическими скоростью вычислений (более миллиарда операций в секунду) и емкостью памяти, превосходящей человеческую. Техническая кибернетика достигла огромных успехов, но машина пока не стала чемпионом мира по шахматам, переводчики не опасаются конкуренции с ее стороны, а сочинение стихов на

ЭВМ по-прежнему остается способом развлечения программистов.

Сегодня не спорят, может ли мыслить ЭВМ, так как понятие «мыслить» является первичным и нельзя дать его точного определения. Однако крушение надежд на быстрое создание искусственного интеллекта связано не с отсутствием этого определения, а со скучными познаниями человечества о материальных основах мыслительных процессов. Поэтому не моделируют мышление, а вынуждены ограничиться его имитацией, рассматривая человеческий мозг как «черный ящик» с неизвестным устройством, свойства которого приходится определять по реакциям на внешние воздействия. Установлено, в частности, что деятельность правого полушария мозга связана с образами, а левая — с логическими связями между символами.

Приближенно можно выделить три вида мышления — рефлекторное, логическое и эвристическое. Рефлекторная деятельность мозга проявляется в решении человеком задач «не задумываясь», в соответствии с навыками, жестко закрепленными в памяти. Логическое мышление связано с умозаключениями о выборе лучшего пути решения задачи на основании хранящейся в памяти логической информации. Эвристическое мышление в основном определяется интуицией, связанной с подсознательной ассоциативной деятельностью мозга. Этот вид мышления в наибольшей степени определяет творческие возможности человека, проявляющиеся в его способности выбирать или даже изобретать пути решения задачи без их полного перебора. К сожалению, имитация этого вида мышления находится в зачаточном состоянии, хотя с ним в наибольшей мере связана вся проблема искусственного интеллекта.

Напомним, что поиск человеком пути решения задачи относится к конфликтным ситуациям и по определению является игрой. Следовательно, с играми связана как кибернетика в целом, так и проблема искусственного интеллекта. Этим объясняются значительные затраты времени и средств на создание, например, программ для игры ЭВМ в шахматы. В процессе создания и использования игровых программ накапливается опыт имитации логического и эвристического мышлений, необходимый для решения проблемы искусственного интеллекта.

В течение многих десятилетий большинство людей узнавало об успехах кибернетики из книг и журналов, так как дорогостоящие универсальные ЭВМ были доступны лишь узкому кругу специалистов. В последние годы это положение принципиально изменилось в связи с созданием недорогих и, следовательно, доступных широкому кругу пользователей ЭВМ четвертого поколения. Они получили название микро-ЭВМ, так как собраны на микросхемах — элементах с высокой степенью интеграции. К универсальным микро-ЭВМ относятся программируемые микрокалькуляторы и персональные ЭВМ.

В алгоритмических языках программирования и конструкциях стационарных универсальных ЭВМ предусмотрены автоматическая засыпка данных в память и вызов из нее с помощью операторов присваивания. При выполнении программы считывание машиной кодов формулы вида $I = A$ приводит к автоматической засыпке числа A или результата вычислений по выражению A в свободную ячейку памяти, которой в качестве адреса присваивается код идентификатора I . Если же ранее встречавшийся в программе идентификатор записан в правой части

формулы $I = A$, то по коду этого идентификатора из памяти вызывается соответствующее число, над которым и выполняется нужная операция.

Например, при считывании в программе формулы $X1 = 5,67$ в ячейку памяти, которой присваивается код идентификатора $X1$, засыпается число 5,67. При последующем считывании, например формулы $K = X1 + 2$, из ячейки с кодом $X1$ вызывается число 5,67, и результат сложения 7,67 в правой части формулы автоматически заносится в ячейку памяти с адресом, соответствующим коду идентификатора K .

Автоматическая пересылка данных с помощью операторов присваивания существенно упрощает составление программ, но ресурс памяти ЭВМ в этом случае используется недостаточно полно. Между тем в 60-х гг. запоминающие устройства отличались относительно большими размерами и стоимостью, что существенно ограничивало емкость памяти малогабаритных ЭВМ.

Успехи полупроводниковой технологии к середине 70-х гг. привели к существенному уменьшению размеров и стоимости запоминающих устройств, что обеспечило возможность создания недорогих малогабаритных микроЭВМ с алгоритмическими языками программирования, подобными языкам стационарных ЭВМ. Относительно низкая стоимость таких машин, получивших название *персональных*, позволяет использовать их в качестве индивидуальных вычислительных средств без практически непрерывного режима работы, при котором окупаются дорогостоящие стационарные ЭВМ. Однако персональные ЭВМ различных типов также значительно различаются как по стоимости, так и по назначению.

Наиболее совершенные персональные ЭВМ, обычно называемые профессиональными, по основным показателям превосходят большинство стационарных ЭВМ третьего (на интегральных компонентах) поколения, но в связи с относительно высокой стоимостью их используют лишь в служебных целях. Значительно дешевые учрежденческие персональные ЭВМ, снабженные ограниченным набором вспомогательных устройств для ввода, хранения и вывода информации. Наименее дорогие из таких машин широко используются для обучения учащихся средней и высшей школы.

Наиболее низкой стоимостью отличаются персональные ЭВМ, называемые домашними и предназначенные для использования с бытовыми телевизором в качестве дисплея (индикатора) и кассетным магнитофоном в качестве накопителя информации. Несколько дороже в настоящее время автономные карманные и настольные персональные ЭВМ, которые в обозримом будущем заменят программируемые микрокалькуляторы.

Внедрение универсальных микроЭВМ в нашу жизнь знаменует принципиально новый этап развития прикладной кибернетики, так как оказывается возможной массовая и экономически выгодная автоматизация решения задач, ранее доступных лишь человеку. Во всем мире быстро растет производство, прежде всего, домашних ЭВМ, используемых для различного рода игр и решения логических и вычислительных «домашних» задач, а расширяющееся внедрение профессиональных и учрежденческих персональных ЭВМ уже сейчас привело к изменению и рационализации решения множества практических задач.

Глава 2

ВХОДНЫЕ ЯЗЫКИ МИКРО·ЭВМ

Любая универсальная микро-ЭВМ характеризуется двумя основными рабочими состояниями — режимом *программирования* для ввода программы в запоминающие устройства и *программируемым* режимом автоматического выполнения программы. Режим программирования преследует ту же цель, что и обучение человека решению конкретной задачи — закреплению в памяти алгоритма решения. Поэтому в некоторых микро-ЭВМ команда перехода в режим программирования обозначена сокращением LRN от слова learn (обучать).

Средством «обучения» ЭВМ являются языки программирования (входные языки), на которых составляются и вводятся в машину программы. Входные языки различаются алфавитом, образованным определенными символами, допустимыми сочетаниями этих символов, имеющими самостоятельный смысл слов (словарным запасом), и синтаксическими правилами, определяющими допустимый порядок следования слов. Последовательность слов, приводящая к искомому результату работы ЭВМ, образует программу.

Языки программирования предназначены для представления алгоритмов, элементами которых являются шаги, содержащие описания операций (операторы). Поэтому основными структурными элементами программ также являются операторы (называемые также предписаниями, инструкциями или командами), образованные одним или несколькими

шагами, причем шаг алгоритма может представляться несколькими шагами программы. Система внешних команд микро-ЭВМ, вводимых нажатиями клавиш, содержит также служебные команды для управления работой машины, не входящие в программу.

Языки программирования микро-ЭВМ подробно описаны в руководствах по их эксплуатации, поэтому ограничимся лишь необходимыми для дальнейшего изложения сведениями об особенностях таких языков.

В персональных микро-ЭВМ используют различные языки программирования, но наибольшее распространение нашел простой алгоритмический «язык для начинающих» БЕЙСИК, используемый и для программирования «больших» ЭВМ. Алфавит базовой (минимальной) версии этого языка [6] содержит 26 заглавных латинских букв от A до Z, четыре знака препинания (точка, запятая, точка с запятой, апостроф), десятичные цифры от 0 до 9, пять арифметических знаков + (плюс), - (минус), * (умножение), / (деление), \uparrow (возведение в степень), шесть знаков отношений (=, \neq , $<$, \leqslant , \geqslant , $>$), знаки круглых скобок, пробела и возврата каретки печатающего устройства. В ЭВМ отечественного производства для вывода информации алфавит обычно дополнен прописными русскими буквами, не совпадающими по начертанию с латинскими.

Числа в языке БЕЙСИК представляют в естественной (с фиксирован-

ной запятой) и показательной (с плавающей запятой) формах, причем дробную часть отделяют точкой, а целый порядок — символом Е (например, число — 0,0068 можно представить в показательных формах — 6,8Е—4 или — 0,68Е—3). Идентификаторы (имена переменных) составляют из латинской буквы, которую можно дополнить цифрой. Разрядность мантиссы и порядка чисел определяется конструкцией конкретной микро-ЭВМ.

Словарный запас языка содержит команды-операторы вычисления синуса SIN (x), косинуса COS (x), тангенса TAN (x), арктангенса ATN (x), натуральных логарифма LOG (x) и антилогарифма или экспоненты EXP (x), квадратного корня SQR (x), модуля ABS (x), целой части INT (x) и знака числа SGN (x), а также команды RND для выборки случайного числа с равномерным распределением в интервале (0,1). В круглых скобках указывают идентификатор аргумента x или определяющее его выражение.

Программа на языке БЕЙСИК состоит из последовательности операторов, записываемых с новой строки и начинающихся числовой меткой m (обычно выбирают m = 10, 20, 30, ..., 999) и следующим за ней именем оператора, обычно дополняемым определенным набором слов. Метки предназначены для адресации переходов и упорядочения машиной последовательности выполнения операторов.

Основным является оператор присваивания m LET I₁ = I₂ = ... = I_n = A, где после имени LET (пусть) записываются идентификаторы I_i и выражение (число, идентификатор или расчетная формула) A. После выполнения этого оператора число A, результат вычислений по формуле A или вызванное из памяти значение идентификатора A заносятся в ячейки

памяти с кодами идентификаторов I_i. Операцию присваивания реализуют также с помощью операторов, называемых DATA (данные) и READ (читать). В первом из них m DATA a₁, a₂, ..., a_n после имени оператора записывают последовательность (массив) чисел, отделяемых запятыми. Во втором операторе m READ I₁, I₂, ..., I_k после имени записывают последовательность идентификаторов, число которых может быть меньше числа элементов массива данных в операторе DATA. При считывании машиной оператора READ в ячейки памяти с кодами его идентификаторов заносятся соответствующие числа из массива данных оператора DATA, которые могут последовательно считываться несколькими операторами READ. Для восстановления массива данных используют оператор m RESTORE (восстановить).

В условном операторе m IF A₁ П A₂ THEN m_n после имени IF (если) записывают условие сравнения выражений A₁ и A₂ на выполнение отношения П, а после слова THEN (тогда) — метку перехода при выполнении проверяемого условия (в противном случае выполняется следующий по порядку метки оператор). Безусловный переход реализуется с помощью оператора m GO TO m_n, где после имени GO TO или GOTO (идти на) записывается метка m_n оператора, выполняемого следующим.

Для вычислений в цикле его начинают оператором m FOR I == A₁ TO A₂ STEP A₃, где после имени FOR (для) записана формула присваивания переменной I численного значения A₁, равного начальному, после слова TO (до) — выражение A₂, определяющее конечное значение переменной, а после слова STEP (шаг) — приращение A₃ переменной на каждом шаге. Оканчивается цикл оператором m NEXT I с именем NEXT

(следующий), проверяющим условие выхода из цикла $I = A_2$ и при его выполнении передающим управление следующему по порядку меток оператору.

Вывод информации обеспечивается оператором m PRINT (печатать), после имени которого записывают идентификаторы переменных, численные значения которых выводятся на печать или индикатор (дисплей), и отделяемый апострофами выводимый текст.

Прекращение выполнения программы обеспечивается операторами m STOP, которые могут записываться в любом месте программы, но она обязательно оканчивается оператором m END (конец) с наибольшей меткой.

Переход в режим программирования обычно обеспечивается командой NEW (новое), пуск программы — командой RUN (пуск). Для повторения вычислений с новыми исходными данными, а также для исправления ошибок и дополнения программы новыми операторами с промежуточными метками после выполнения программы вводят нужный оператор и команду RUN.

Составим, например, программу используемого при решении многих игровых задач преобразования позиционных представлений чисел при изменении основания m системы счисления. Напомним, что такие абстрактные математические объекты, как числа в позиционных системах счисления, подобных обычной десятичной или рассмотренной двоичной, представляют последовательностями $a_r \dots a_2 a_1 a_0$ позиций, называемых m -ичными разрядами и содержащих по одной цифре из множества $0, 1, \dots, m - 1$. В крайнем правом разряде цифра отображает соответствующее натуральное число, но в каждом левом разряде той же цифрой отображается в m раз большее число, чем в соседнем правом разряде.

Представления чисел N_p и N_q в системах счисления с основаниями p и q связаны отношением

$$N_q = a_r p^r + \dots + a_2 p^2 + a_1 p^1 + a_0 p^0,$$

где $a_{i+1} = N_p - qN_{pi+1}$ и N_{pi+1} — соответственно остаток и целая часть частного от деления N_{pi} на q при $i = 0, 1, \dots, r$; $N_{p0} = N_p$.

Преобразование представления N_p в представление N_q в соответствии с этим отношением реализуется алгоритмом [12]:

1. Принять $i = 0; p^i = 1; S_{i+1} = 0; N_{pi} = N_p$.
2. Вычислить N_{pi}/q .
3. Определить целую часть частного $N_{pi+1} = E[N_{pi}/q]$.
4. Определить целую часть частного $N_{pi+1} = E[N_{pi}/q]$.
5. Вычислить остаток $a_{i+1} = N_{pi} - qN_{pi+1}$.
6. Вычислить сумму $S_{i+1} = S_i + a_{i+1}p^i$ и $p = pp^i$.
7. Если $N_{pi+1} = 0$, то перейти к шагу 8, иначе — к шагу 7.
8. Принять $i = i + 1$ и перейти к шагу 2.
9. Принять $N_q = S_{i+1}$ и закончить решение задачи.

На языке БЕЙСИК этот алгоритм при обозначении исходных данных символами p , q и N_p можно представить программой:

```
NEW
10 DATA p, q, Np
20 READ P, Q, NP
30 DATA 1, 0
40 READ K, S
50 LET B = N
60 LET A = INT(B/Q)
70 LET S = S + (B - A * Q) * K
80 LET K = K * P
90 LET B = A
100 IF B = 0 THEN 120
110 GO TO 60
120 PRINT 'N('P') = 'N' ПРЕОБРАЗОВАНО
B N('Q') = 'S
130 END
RUN
```

После выполнения этой программы при исходных данных $p = 3, q = 10, N_p = 121212$ будет выведен результат:

$N(3) = 121212$

ПРЕОБРАЗОВАНО В $N(10) = 455$

Для повторного выполнения программы (например, при $p = 10, q = 8$ и $N_p = 455$) следует ввести с клавиатуры:

```
10 DATA 10, 8, 455
RUN
```

В этом случае получим следующий результат:

$N(10) = 455$ ПРЕОБРАЗОВАНО В $N(8) = 707$

В персональных ЭВМ различных типов часто используют расширенные версии языка БЕЙСИК с дополнительными операторами (например, MAT для операций над матрицами или GOSUB для обращений к подпрограммам) и командами для обмена информацией с периферийными

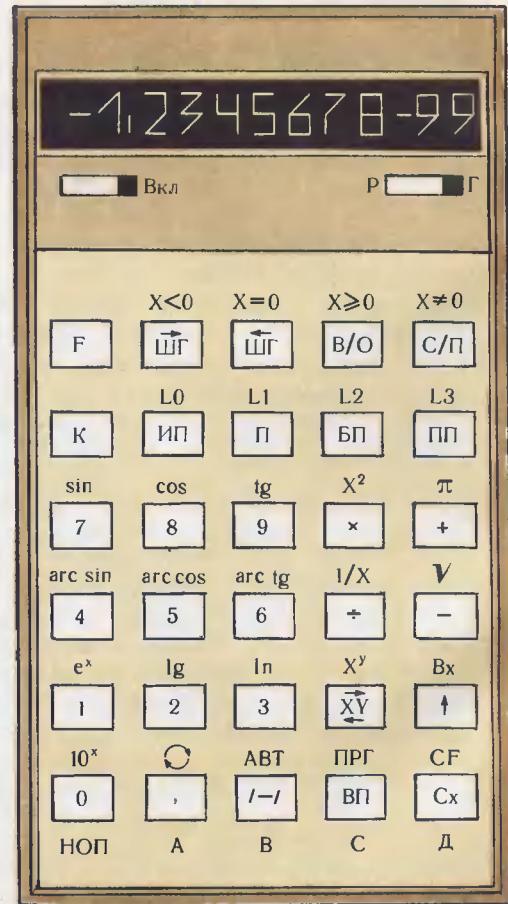
устройствами и вывода таблиц и графиков на дисплей. В большинстве персональных ЭВМ предусмотрена также возможность хранения в накопителях информации программных трансляторов для ввода в память ЭВМ программ, составленных на других языках программирования.

Особенности входных языков программируемых микрокалькуляторов рассмотрим для распространенной массовой микро-ЭВМ «Электроника Б3-34» (рис. 5, а), а также отличающейся от нее лишь источником автономного питания микро-ЭВМ «Электроника Б3-54», карманной «Электроника МК-54» и настольный «Электроника МК-56» (рис. 5, б). Эти микрокалькуляторы снабжены числовой памятью из 14 адресуемых регистров с шестнадцатиричными номерами 0, 1, ..., 9, А, В, С, Д и программной памятью из 98 ячеек для хранения 8-битовых шагов программы. Программная память разбита на 10 страниц по 10 ячеек (8 на последней странице) с десятичными адресами ab , где $a = 0, 1, \dots, 9$ — номер страницы и $b = 0, 1, \dots, 9$ — номер ячейки на странице. По адресам ячеек реализуются переходы к соответствующим шагам программы.

Индикатор содержит 12 знакомест из семи световых сегментов для выравнивания десятичных представлений чисел и сигнала ошибки ЕГГОГ в рабочем режиме, а также шестнадцатиричных кодов шагов программы и адреса вводимого следующим шага в режиме программирования. Диапазон представления абсолютных значений чисел — от $1 \cdot 10^{-99}$ до $9,9999999 \cdot 10^{99}$, причем на отрезке от 1 до 99999999 результаты операций выравниваются в естественной форме, а в остальной части диапазона — в показательной форме $a = m \cdot 10^n$, где m — восьмиразрядная мантисса с запятой после первой

значающей цифры ($1 \leq M < 10$), а n — двухразрядный целый порядок.

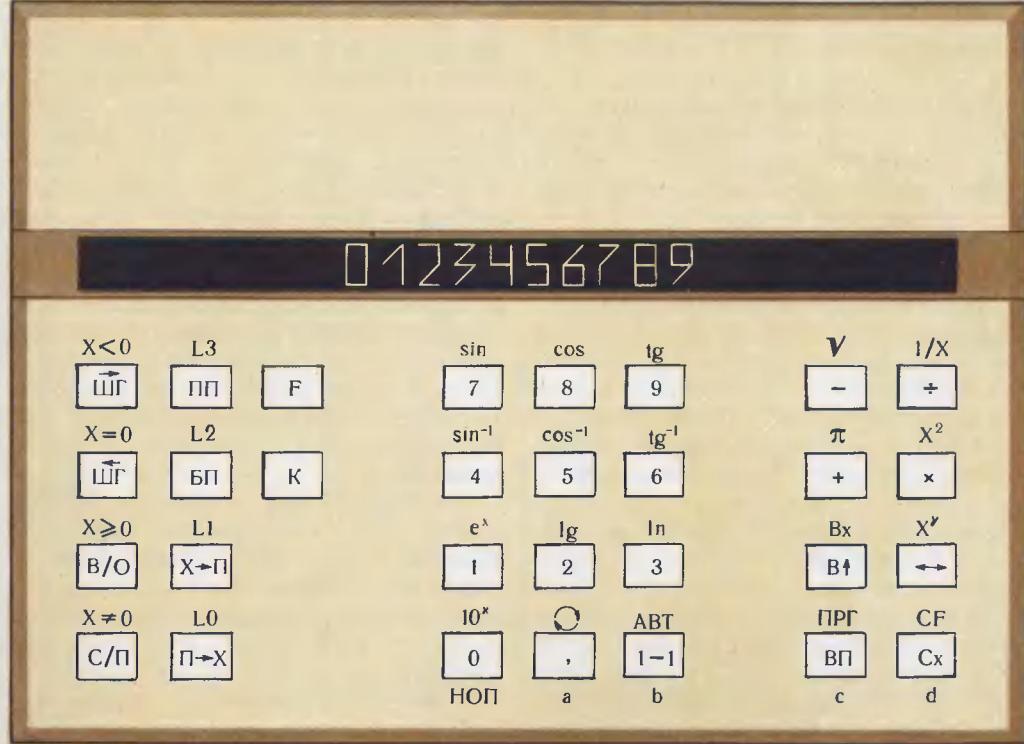
Процессор содержит операционный стек с регистрами X, Y, Z, T и вспомогательным регистром $X1$, а также арифметико-логическое устройство для выполнения по программе операций над содержимым регистров X и Y . Входной регистр X соединен с индикатором через дешифратор, преобразующий двоичное содержимое регистра в напряжение, подаваемые на сегменты знакомест индикатора для выравнивания соответствующих символов.



Алфавит языка программирования микрокалькуляторов образован указанными на клавиатурах символами команд, за исключением служебных команд $\overrightarrow{\text{ШГ}}$, $\overleftarrow{\text{ШГ}}$, АВТ, ПРГ, F, CF, не используемых при выполнении программ. Имеются незначительные орфографические различия в символах команд — обозначенные на клавиатуре микрокалькуляторов «Электроника Б3-34» и «Электроника Б3-54» символы П, ИП, \uparrow , XY, arcsin, arccos и arctg (рис. 5, а) отображены на клавиатуре микрокалькуляторов «Электроника МК-54» и «Электроника МК-56» соответственно символами $x \rightarrow \Pi$, $\Pi \rightarrow x$, B \uparrow , \leftrightarrow , \sin^{-1} , \cos^{-1} и tg^{-1} (рис. 5, б). В дальнейшем для определенности будем использовать

только первые из этих символов как более удобные, а для упрощения записи команды обмена содержимым регистров X и Y и поворота стека (обозначенной окружностью со стрелками) будем обозначать соответственно символами XY и \rightarrow .

Для уменьшения количества клавиш часть команд обозначена над клавишами, и для их ввода нажимают префиксные клавиши K (для ввода операторов косвенной адресации и команды НОП) и F. Так как эти клавиши не имеют отношения к алгоритмам, то в программах будем сохранять лишь символ K для операторов косвенной адресации, а в остальных случаях символы K и F не будем указывать (как не указывают



символы поднятия каретки в машино-писных текстах).

Коды чисел в регистре X набирают командами набора цифр от 0 до 9, десятичного разделительного знака и числа $\pi = 3,141\ 592\ 6$, а также командами $/$ — $-$ — изменения знака, ВП — ввода порядка. Вызов в регистр X копии содержимого регистра N памяти и засылка копии содержимого регистра X в регистр N памяти обеспечиваются соответственно операторами обращения к памяти ПН и ИПН, вводимыми нажатиями клавиш П. или ИП и клавиш с номером регистра N .

Для вычислений служат двухместные (выполняемые над двумя операндами) функциональные операторы $+$, $-$, \times , \div , X^y и одноместные операторы \sin , \cos , \tg , \arcsin , \arccos , \arctg (размерность аргумента устанавливается переключателем Р — Г или Р — ГРД — Г), x^2 , \sqrt{x} , $1/x$, \ln , e^x , \lg , 10^x . Последовательность ввода операторов и операндов и соответствующие правила работы операционного стека определяются синтаксическими правилами входного языка.

Синтаксические правила выполнения вычислений в языке БЕИСИК и большинстве других языков программирования, включая входные языки некоторых программируемых микрокалькуляторов, соответствуют обычной алгебраической записи формул, при которой операторы вводят после первого операнда (например, $\sin x$ или $x + y =$). В большинстве входных языков программируемых микрокалькуляторов, включая и рассматриваемые, использованы синтаксические правила, соответствующие записи функциональных операторов после operandов (например, $x \sin$ или $x \uparrow y +$), называемой обратной записью. Такие правила существенно упрощают конструкцию микрокалькуляторов, так как не требуется

предварительное запоминание кодов операторов, и фактически соответствуют выполнению человеком арифметических операций над многозначными числами, когда вначале записываются эти числа, а затем выполняется операция.

В соответствии с этими правилами при наборе или вызове из памяти число заносится в регистр X , а предыдущее содержимое регистров стека смешается «вверх» (рис. 6, а). При вводе оператора \uparrow содержимое стека смешается вверх с сохранением содержимого регистра X (рис. 6, б). При наборе числа после оператора \uparrow или оператора стирания Сх оно заносится в регистр X без изменения содержимого остальных регистров (рис. 6, в). При вводе оператора XY прежнее содержимое регистра X засыпается в регистры Y и $X1$, а регистра Y — в регистр X (рис. 6, г). Ввод оператора \rightarrow поворота стека смешает его содержимое «вниз», причем прежнее содержимое регистра X засыпается в регистры T и $X1$ (рис. 6, д). Ввод оператора Вх смешает стек «вверх» с засыпкой содержимого регистра $X1$ в регистр X (рис. 6, е). После выполнения одноместного оператора результат заносится в регистр X , прежнее содержимое которого засыпается в регистр $X1$ (рис. 6, ж), а после выполнения двухместного оператора содержимое стека дополнительно смешается «вниз» (рис. 6, з).

Рассмотренные операторы используют как в программах автоматических вычислений, так и при вычислениях нажатиями клавиш. Специфическими для программ автоматических вычислений являются занимающие по одному шагу программы операторы управления программой С/П (стоп/пуск), В/О (возврат/очистка) и косвенной адресации КПН, КИПН, КБПН, КППН, KxSON (где

буквой S обозначен знак отношения $=$, \neq , \geqslant , или $<$). Операторы КПН и КИПН обеспечивают обращение к регистру памяти, номер которого равен модифицированному содержимому адресного регистра N. Аналогично операторы косвенных переходов КБПН, КППН и KxS0N обеспечивают переход по адресу, равному модифицированному содержимому адресного регистра N. При каждом выполнении таких операторов содержимое регистра N модифицируется — отбрасывается дробная часть, целая часть уменьшается на единицу, если $N \leqslant 3$; увеличивается на единицу,

если $N = 4\dots6$, и не изменяется при $N > 6$.

В программах автоматических вычислений используют также операторы (занимающие по два шага программы) безусловного перехода БП ab, обращения к подпрограмме ПП ab, четыре условных оператора вида xS0 ab и четыре оператора цикла вида LN ab, где $N \leqslant 3$, а ab — адрес перехода.

Особенности программирования на входных языках микрокалькуляторов подробно описаны в руководствах по их эксплуатации и книгах [9, 10, 12, 13]. Для записи программ обычно

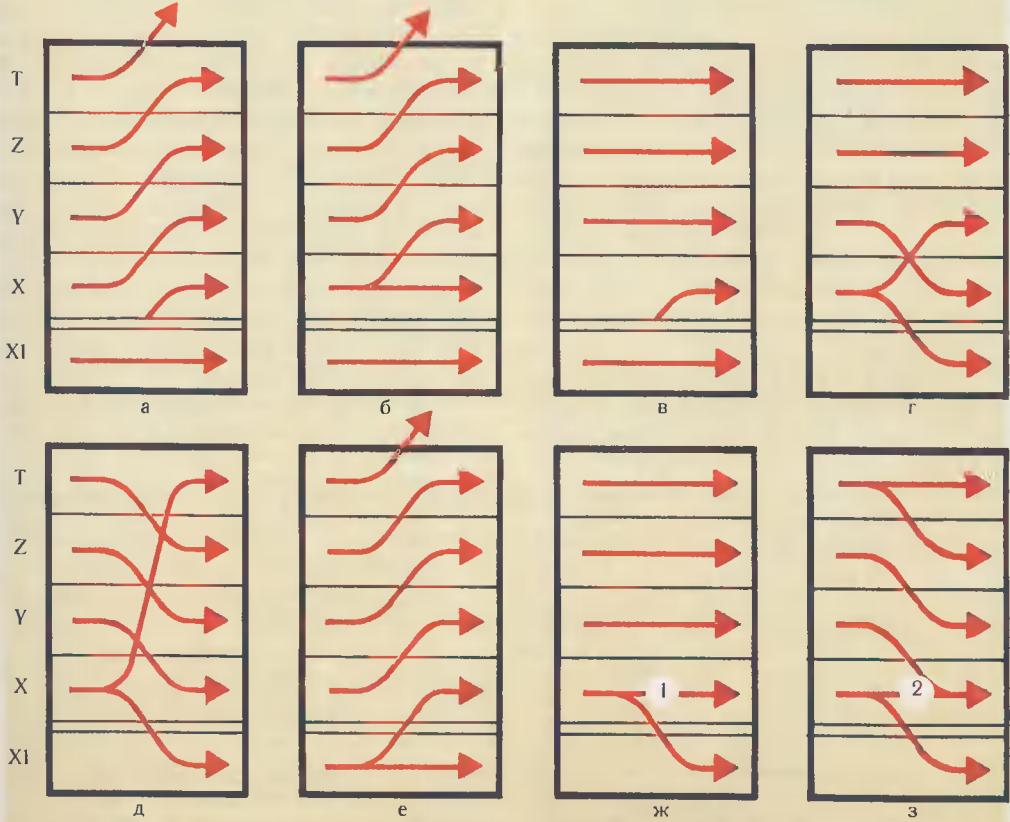


Рис. 6.

используют таблицы с указанием для каждого шага нажимаемых клавиш, высвечиваемого кода и адреса шага. Такая табличная форма удобна при изучении программирования и записи готовых программ для их хранения, но занимает много места и практически непригодна для составления программ, когда их текст приходится многократно переписывать в поисках наилучшего варианта. В последнем случае удобно использовать компактную запись программы по 10 шагов в строку, так как в этом случае адрес любого шага легко определить по номерам $a = 0, 1, \dots, 9$ строки и $b = 0, 1, \dots, 9$ столбца, на пересечении которых находится этот шаг.

Для удобства контроля правильности ввода в программную память постоянно хранимых (библиотечных) программ в их компактной записи каждый шаг целесообразно дополнять кодом, высвечиваемым на индикаторе в режиме программирования. Примером может служить программа 1, обеспечивающая решения тех же задач, что и ранее составленная программа на языке программирования БЕЙСИК.

В инструкциях к выполнению программы должны содержаться описания всех действий пользователя после

ввода программы и перехода в рабочий режим нажатием клавиш с обозначениями F и АВТ. При составлении программ целесообразно использовать компактную запись инструкций, обозначая операции ввода исходных данных формулами вида $a = PN$, где a — набираемое на индикаторе число, а PN — содержимое регистра N памяти или операционного стека, куда следует занести число a (при записи вида $a = PX$ набранное число перед пуском программы должно храниться в регистре X). Размещение результатов выполнения программы в регистрах операционного стека и памяти удобно отображать формулами вида $PN = a$ (при записи $PX = a$ результат a выполнения программы хранится в регистре X и высвечивается на индикаторе). Кроме того, в компактной записи инструкции должны быть указаны клавиши, нажимаемые для пуска программы. Например, инструкция к программе 1 в компактной записи будет иметь следующий вид: ($p = P8, q = P9$) $N_p = = PX$ В/О С/П $PX = N_q$, где круглыми скобками обозначены исходные данные, сохраняющиеся после выполнения программы.

Инструкции к библиотечным программам целесообразно дополнять

Программа 1. Преобразование позиционных представлений чисел N_p и N_q с основаниями $p < 10$, $q = 10$ или $p = 10$, $q < 10$

Х	Y	Сх	0Г	П7	47	1	01	Вх	0	↑	0Е	ИП9	69	÷	13	1	01	+	10
П0		КИП0	Г0	→	25	→	25	ИП0	60	ИП9	69	×	12	—	11	×	12	ИП7	67
+	10	П7	67	→	25	ИП8	68	×	12	ИП0	60	$x = 0$	5Е	05	05	ИП7	67	С/П	50

Инструкция. Ввести числа p и q соответственно в регистры 8 и 9 (эти числа сохраняются после выполнения программы, и при повторном ее выполнении с теми же числами p и q их можно не вводить) и число N_p в регистр X , нажать клавиши В/О и С/П,

после выполнения программы искомое число N_q высвечивается на индикаторе.

Время счета в секундах около $3 + 6r$, где r — большее число разрядов в представлениях чисел N_p и N_q .

контрольными примерами для проверки правильности ввода и выполнения программ, а в нужных случаях указывать и время их выполнения.

В программируемых микрокалькуляторах «Электроника МК-61» (рис. 7) и «Электроника МК-52» имеется 15 адресуемых регистров числовой памяти с шестнадцатеричными номерами 0, 1, ..., 9, A, B, C, D, E и программная память емкостью 105 шагов, причем переходы в программе допускаются только к шагам 00...99. Кроме того, в микрокалькуляторе «Электроника МК-52» имеется электрически перепрограммируемое запоминающее устройство емкостью 4 Кбит (512 шагов программы) для длительного хранения программ и числовых данных при выключении питания, а также предусмотрена возможность использования блоков расширения памяти для постоянного хранения прикладных программ и других периферийных устройств.

Входной язык этих микрокалькуляторов отличается от языка программирования микрокалькулятора «Электроника Б3-34» лишь дополнительными операторами (вводимыми с нажатием префиксной клавиши K) для определения целой и дробной частей, знака и модуля чисел, большего из двух чисел, выполнения операций логического сложения и умножения, исключающего ИЛИ и инверсии, а также команд преобразования часов в часы, минуты и секунды, градусов в градусы и минуты и обратно. Кроме того, имеется оператор СЧ для вызова квазислучайных чисел с равномерным распределением в интервале (0; 1).

Использование этих операторов позволяет сократить длину и время выполнения многих программ, составленных на входном языке микрокалькулятора «Электроника Б3-34»

и его аналогов. Например, при использовании оператора [x] для выделения целой части содержимого регистра X программа 1 сокращается на пять шагов:

XY Cx П7 1 Bx ↑ ИП9 ÷ [x] П6

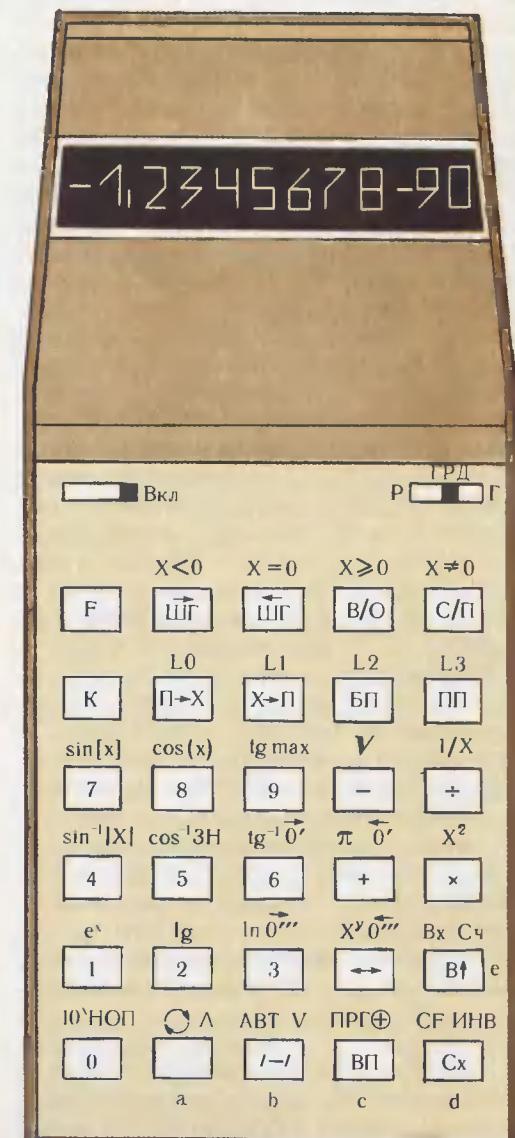


Рис. 7.

ИП9 \times — \times ИП7 + П7 \rightarrow ИП8
 \times ИП6 $x = 0$ 05 ИП7 С/П

Входные языки программируемых микрокалькуляторов с условными обозначениями операторов внешне напоминают языки ассемблеров, но в действительности они являются языками программирования высокого уровня, обеспечивающими непосредственное отображение шагов алгоритмов. Основные отличия этих языков от обычных алгоритмических языков, подобных языкам БЕЙСИК или ФОРТРАН, заключаются в способе обращения к памяти, неформальности операторов присваивания и компактности записи программ. Так, для ввода условного оператора $x = 0 ab$ требуется четыре нажатия клавиш (F , $x = 0$, a , b), тогда как для ввода аналогичного оператора $m IF x = = THEN m_n$ на языке БЕЙСИК требуется нажать значительно большее число клавиш. Для ввода оператора цикла $LN ab$ также требуется нажать только четыре клавиши, а для ввода аналогичного цикла в программу на языке БЕЙСИК необходимо записать в начале цикла оператор $m FOR I = n TO 0 STEP 1$ и в конце $m NEXT I$.

Компактность записи программ на языках микрокалькуляторов обеспечивается и неформальной записью операторов присваивания без ключевых слов LET или ПУТЬ. Например, фрагмент программы «... + ИП6 \div \div ln...» при обозначении содержимого регистров X и Y буквами x и y соответствует оператору присваивания «пусть $x = \ln(x + y / ИП6)$ » или последовательности операторов «пусть $x = x + y$ », «пусть $x = x / ИП6$ », «пусть $x = \ln x$ ».

Различие в способах обращения к памяти, обеспечивающих полное использование памяти микрокалькуляторов, также оказывается не очень существенным, если рассматривать операторы ПН и ИПН как операторы

присваивания «пусть $PN = x$ » и «пусть $x = PN$ », где символ содержащего регистра памяти PN является идентификатором (именем) переменной, численное значение которой хранится в этом регистре.

Рассмотренные особенности входных языков программируемых микрокалькуляторов, связанные с ограниченной емкостью запоминающих устройств, к сожалению, затрудняют чтение программ и понимание представленных ими алгоритмов. Однако компактность программ на этих языках и возможность полного использования ресурса памяти во многих случаях являются преимуществами и при программировании микро-ЭВМ других классов. Поэтому, например, для настольной микро-ЭВМ «Электроника ДЗ-28» предусмотрена возможность ввода программ как на языке БЕЙСИК, так и на входном «микрокалькуляторном» языке. Подобный язык находится и среди трех «собственных» языков различного уровня персональной («домашней») ЭВМ «Электроника МК-72». Следует также отметить, что у многих микро-ЭВМ операторы входного языка, подобного языку БЕЙСИК, вводят не набором отдельных знаков, а нажатием клавиш (например, GOTO, LET или IF), как и для программируемых микрокалькуляторов.

В настоящей книге в связи с разнообразием входных языков микро-ЭВМ программы, иллюстрирующие алгоритмы решения логических и игровых задач, приведены на входном языке распространенного микрокалькулятора «Электроника Б3-34» и его аналогов и полностью пригодны для программирования микрокалькуляторов «Электроника МК-61» и «Электроника МК-52», хотя в последнем случае многие из приведенных программ могут быть сокращены.

Глава
3

РЕШЕНИЕ ЛОГИЧЕСКИХ ЗАДАЧ

Поиск человеком решения новой задачи, как уже отмечалось, по определению является игрой. Это, прежде всего, игра с природой, в которой умения и знания человека преодолевают непредвиденные трудности — будь-то чрезмерная погрешность вычислений, определяемая законами математики, невозможность реализации выбранного способа решения по физическим законам или неучтенное влияние случайных событий.

Человек находит наилучший (оптимальный) путь решения задачи с помощью логических умозаключений, но искомые результаты могут быть как количественными оценками, так и требуемыми отношениями между некоторыми объектами. Вычисления, приводящие к количественным оценкам, основаны на формальной (математической) логике, но при вычислениях она не проявляется так явно, как при поиске требуемых отношений.

Поэтому задачи последнего вида будем условно называть логическими.

При решении логических задач с помощью ЭВМ, предназначенных для операций над числами, также приходится прибегать к вычислениям, но они обычно лишь обеспечивают проверку отношений между некоторыми объектами, символами которых служат порядковые (целые) числа. Поэтому при решении многих логических задач явно или неявно используется основная формула теории сравнений $a \equiv b(m)$. Эта формула читается « a и b сравнимы по модулю

m » и означает, что при делении целых чисел a и b на положительное целое число m образуются одинаковые остатки и, следовательно, частное $(a - b)/m$ равно целому числу.

Простейшим примером приложения теории сравнений является поиск целочисленных решений уравнений с несколькими неизвестными, называемых диофантовыми и имеющих в общем случае бесконечное число решений. Диофантово уравнение $ax + by = c$, имеющее бесконечное число решений, в конечном интервале изменения переменных x и y при выполнении условия сравнения $x \equiv by/a$ содержит конечное число целочисленных решений $x = (c - by)/a$.

Особенности диофантовых уравнений используют в различных играх и логических задачах. Например, один из участников подобной игры для выбранных им дня $D = 1, 2, \dots, 31$ и месяца $M = 1, 2, \dots, 12$ указывает число $A = 12D + 31M$, а другой участник игры по этому числу должен определить исходные числа D и M . Для этого он должен последовательно подставлять числа $M = 1, 2, \dots$ в правую часть формулы $X = (A - 31M)/12$ до получения целого числа $X = D$. Для программирования микро-ЭВМ этот способ решения задачи отображается следующим алгоритмом:

1. Принять $M = 1$.
2. Вычислить $X = (A - 31M)/12$.
3. Определить дробную часть $D(X)$ числа X .
4. Если $D(X) = 0$, то перейти к шагу 6, иначе — к шагу 5.

5. Принять $M = M + 1$ и перейти к шагу 2.

6. Принять день $D = X$ и месяц M .

На используемом входном языке программируемых микрокалькуляторов этот алгоритм реализуется программой 2, после выполнения которой на индикаторе высвечиваются числа D и M , отделенные запятой. Так, при минимальном числе $A = 43$ высвечивается 1,01 или 1 января (время счета около 10 с), а при максимальном числе $A = 744$ высвечивается 31,12 или 31 декабря (время счета около 65 с).

Теория сравнения применима и для автоматизации такой трудоемкой логической задачи, как составление различных расписаний. Простейшим примером может служить составление расписания спортивных встреч m участников (скажем, шахматистов или футбольных команд) в круговом турнире, который при встрече «каждого с каждым» содержит $m - 1$ туров.

При четном числе m участников (если действительное число участников нечетно, то добавляется m -й фиктивный участник, встреча с которым в расписании соответствует свободному дню) каждый n -й участник ($n = 1, 2, \dots, m - 1$) в каждом k -м туре встречается с участником с номером $y_{nk} \neq n$. Методами теории сравнений можно установить, что для $n \leq m - 1$ при $k \leq n$ в k -м туре номер участника

$y_{nk} = m$, если $m - 1 + k = 2n$ и $y_{nk} = m - 1 + k - n$ в остальных случаях, а при $k > n$ номер $y_{nk} = m$, если $k = 2n$ и $y_{nk} = k - n$ в остальных случаях. Построение расписания согласно этим соотношениям отображается следующим алгоритмом:

1. Принять $n = k = 1$.
 2. Если $n - k \geq 0$, то перейти к шагу 3, иначе — к шагу 5.
 3. Если $m - 1 + k - 2n = 0$, то перейти к шагу 7, иначе — к шагу 4.
 4. Принять $y_{nk} = m - 1 + k - n$ и перейти к шагу 8.
 5. Если $k - 2n = 0$, то перейти к шагу 7, иначе — к шагу 6.
 6. Принять $y_{nk} = k - n$ и перейти к шагу 8.
 7. Принять $y_{nk} = m$.
 8. Если $k + 1 - m = 0$, то перейти к шагу 9, иначе принять $k = k + 1$ и перейти к шагу 2.
 9. Если $n = m - 1$, то закончить составление списка для встреч команд $n \leq m - 1$, иначе принять $n = n + 1$, $k = 1$ и перейти к шагу 2.
- Этот алгоритм для $n \leq m - 1$ реализуется программой 3.
- После каждого выполнения этой программы высвечиваемый номер y_{nk} следует записать на пересечении строки n и столбца k расписания и после записи всех элементов y_{nk} строки ввести новое значение $n = 1, 2, \dots, m - 1$ и повторить выполнение инструкции (m вводят только перед началом составления расписания). При

Программа 2. Определение дня D и месяца M по заданному числу $A = 12D + 31M$

П9 49	Сx 0Г	П6 46	КИП6 Г6	ИП9 69	ИП6 66	3 03	1 01	× 12	— 11
1 01	2 02	÷ 13	П8 48	ИП8 68	КИП8 Г8	→ 25	ИП8 68	— 11	x = 0 5Е
03 03	ИП6 66	1 01	0 00	0 00	÷ 13	ИП8 68	+	10	C/П 50

Инструкция. $A = PX$ В/О С/П $PX = D.M.$

Программа 3. Составление расписания встреч m участников кругового турнира

·П7 47	Сх 0Г	П6 46	КИП6 Г6	ИП7 67	ИП6 66	—	$x \geq 0$ 59	13	13	1 01
+ 10	ИП9 69	— 11	/—/ 0L	ИП8 48	ИП7 67	— 11	$x = 0$ 5E	21	21	ИП9 69
П8 48	ИП8 68	С/П 50	БП 51	03	03					

Инструкция. (Четное $m = P9$) $n = RX$
 В/О С/П $PX = y_{n1}$ С/П $PX = y_{n2} \dots$ С/П $PX =$
 $= y_{n, m-1}$.

нечетном числе $m - 1$ участников номер $y_{nk} = m$ означает, что в k -м туре n -й участник свободен от встреч. Если число участников турнира четное и равно m , то для определения элементов y_{mk} строки для участника $n = m$ достаточно в каждом k -м столбце по номеру $y_{nk} = m$ найти номер n соответствующей строки и принять $y_{mk} = n$ (табл. 1).

1. Расписание встреч $m = 8$ участников кругового турнира

n	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$	$k = 7$
1	7	8	2	3	4	5	6
2	6	7	1	8	3	4	5
3	5	6	7	1	2	8	4
4	8	5	6	7	1	2	3
5	3	4	8	6	7	1	2
6	2	3	4	5	8	7	1
7	1	2	3	4	5	6	8
8	4	1	5	2	6	3	7

Программа 3 пригодна для составления расписаний турниров с любым числом участников. Если оно невелико, то целесообразно упростить составление расписания, автоматизировав ввод очередного номера n (для чего фрагмент, подобный программе 3, следует охватить циклом с $m - 1$ итерациями) и выводя после каждого выполнения программы множество всех элементов очередной n -й строки. Если $m \leq 8$, то для формирования

такого множества номеров можно воспользоваться десятичными разрядами представления чисел в регистрах программируемого микрокалькулятора с восьмиразрядной мантиссой. Эта задача решается с помощью фрагмента вида ИПН₁ ИПН₂ 1 0 × × + ПН₂, после выполнения которого содержимое $y_{n1} \dots y_{ni}$ разрядов регистра N_2 сдвигается на один разряд влево, в освободившийся разряд заносится содержимое y_{ni+1} регистра N_1 и результат $y_{n1} \dots y_{ni} y_{ni+1}$ операций заносится в регистр N_2 .

Для формирования множества номеров y_{mk} последней m -й строки расписания для четного числа участников целесообразно использовать регистры 7, 8, ..., Д (регистры с меньшими номерами удобно использовать в качестве адресных регистров операторов косвенного обращения к памяти при организации автоматического увеличения номеров n и k), сопоставив их с номерами туров $k = 1, 2, \dots, 7$. После выполнения цикла операций для каждого значения n и проверки равенств $y_{nk} = m$ в $(k + 6)$ -й регистр заносится номер n и после окончания циклов для всех $n \leq m - 1$ с помощью оператора косвенного вызова из памяти и рассмотренного фрагмента на индикатор можно вывести искомую последовательность y_{mk} для $n = m$. Эти приемы реализованы в программе 4, после каждого выполнения которой (перед составлением расписа-

Программа 4. Составление расписания встреч $m \leq 8$ участников кругового турнира

Сх 0Г	П4 44	КИП4 Г4	Сх 0Г	П2 42	П5 45	КИП5 Г5	ИП4 64	ИП5 65	— 11
$x \geq 0$ 59	16 16	1 01	+	10	ИП0 60	— 11	/ — / 0L	П1 41	ИП4 64
$x = 0$ 5E	30 30	ИП0 60	П1 41	ИП5 65	7 07	+	10	П3 43	ИП4 64
ИП1 61	ИП2 62	1 01	0 00	×	12	+	10	П2 42	ИП0 60
1 01	— 11	$x = 0$ 5E	06 06	ИП2 42	ИП4 44	С/П 50	БП 51	02 02	Сх 0Г
П2 42	6 06	П6 46	КИП6 Г6	ИП2 62	1 01	0 00	×	12	+
ИП6 66	1 01	3 03	— 11	$x = 0$ 5E	53 53	ИП2 62	ИП0 60	С/П 50	П2 42

Инструкция. Очистить регистры 7, 8, ..., D ; четное число $m = P0$ В/О С/П $PX = 1$; $PY = y_{11} y_{12} \dots y_{1m-1}$ (для вызова содержимого регистра y достаточно нажать клавишу XY)

$C/P PX = 2; PY = y_{21} \dots y_{2m-1} \dots C/P PX = m - 1; PY = y_{m-1,1} \dots y_{m-1,m-1}$ БП 4 9 С/П $PX = m; PY = y_{m1} \dots y_{m,m-1}$.

ния в память вводится только число m) на индикаторе высвечивается очередное множество элементов y_{nk} строки расписания.

Если число участников турнира нечетное и равно $m - 1$ или если (при четном числе участников) элементы m -й строки расписания определяются по ранее составленным строкам «вручную», то программу 4 можно существенно сократить, изъяв операторы ИП5 ... КПЗ с адресами от 24 до 29 и фрагмент программы после оператора БП 02, начинающийся с адреса 49, а также заменив оператор $x = 0$ 30 на $x = 0$ 24. В этом случае регистры 7, ..., D перед составлением расписания можно не очищать, а выполнение инструкции следует оканчивать после высвечивания и регистрации множества элементов $y_{m-1,k}$ для ($m - 1$)-й строки расписания.

Расписание, составленное с помощью программ 3 или 4, не является единственным возможным и легко

преобразуется в множество других вариантов. Всего допустимо $(m - 1)!$ перестановок столбцов расписания, соответствующих перенумерации турнов, и $(m - 1)!$ для нечетного или $m!$ для четного числа участников способов нумерации участников турнира. Следовательно, составляемое по приведенным программам расписание является лишь одним из $(m - 1)!$ $(m - 1)!$ или $m! (m - 1)!$ вариантов.

Кстати, точное значение факториала $a! = a(a - 1)(a - 2) \dots 2 \cdot 1$ положительного целого числа $a < 11$ и приближенное значение факториала $a < 70$ вычисляется по очень короткой программе:

П0 1 ИП0 × L0 02 С/П с инструкцией $a = RX$ В/О С/П $PX = a!$. С помощью этой программы несложно установить, что приведенное в табл. 1 расписание является одним из $(8!)^2 = 1625702400$ возможных вариантов.

Решение большинства логических задач связано с операциями над

множествами некоторых объектов, сводящихся к операциям над элементами этих множеств. В программах для персональных ЭВМ элементы множеств могут отображаться буквенными или буквенно-цифровыми идентификаторами, а операции над ними — реализовываться с помощью различных стандартных операторов (например, DATA и READ в языке БЕЙСИК). Однако в программах для микрокалькуляторов элементы множеств приходится отображать цифрами и хранить их (по образцу программы 4) в одном или нескольких регистрах памяти. В этих случаях максимальное число элементов множества не может превышать соответственно разрядности мантиссы или числа свободных регистров числовой памяти. Кроме того, во входных языках программируемых микрокалькуляторов отсутствуют специальные операторы для операций над множествами и их приходится формировать из словарного запаса языка программирования.

Если все элементы множества хранятся в одном регистре памяти микрокалькулятора, то для операций над этим множеством приходится последовательно выделять его элементы в виде цифр, содержащихся в отдельных десятичных разрядах. Существуют различные способы реализации такой операции, среди которых можно выделить достаточно удобный, заключающийся в предварительной нормализации множества введением запятой после первого элемента или перед последним. В этом случае с помощью операторов косвенного вызова из памяти отделяют «целую» и «дробную» части нормализованного множества и выделяют один или несколько элементов исходного множества. Например, при нормализации множества делением на 10 его последний элемент окажется отделенным запятой, и после выполнения

фрагмента вида ПН КИПП → ИПП — 1 0× (при $N > 6$) в регистре X будет находиться последний элемент исходного множества, а в регистре N — остальные его элементы.

Операции с выделением и сравнением элементов множеств выполняются, в частности, в распространенных играх с загадыванием слов или многозначных чисел, когда отгадывающий участник игры предлагает очередные варианты, руководствуясь указаниями загадывающего участника об общем количестве угаданных элементов (букв или цифр) и числе угаданных мест их расположения. Цель таких игр заключается в отгадывании слова или многозначного числа за наименьшее число попыток.

Примером подобной игры может служить «компоновка чисел», один участник которой загадывает пятизначное число, а второй предлагает свои варианты этого числа, руководствуясь после каждой попытки сведениями о числе верно угаданных цифр и числе разрядов с угаданными цифрами. Для участия микрокалькулятора * с рассмотренным входным языком в качестве загадывающего участника этой игры необходимо предварительно составить алгоритм, обеспечивающий формирование пятизначного числа, не известного противнику, и сравнение его цифр с цифрами предлагаемых вариантов для определения числа угаданных цифр и числа разрядов с угаданными цифрами. Такой алгоритм можно представить следующим описанием:

1. Сформировать пятизначное число a с цифрами a_i и принять число попыток $k = 0$.

2. Запомнить пятизначное число b , предлагаемое противником.

* Для подобной игры за рубежом был даже выпущен специальный микрокалькулятор СОМР-4.

3. Принять $k = k + 1$, число $n = 0$ угаданных цифр и число $m = 0$ разрядов с угаданными цифрами.

4. Нумеруя разряды a и b слева направо, принять $i = 5$, $j = 6$.

5. Выделить цифры a_i и b_j .

6. Если $a_i = b_j$, то перейти к шагу 7, иначе — к шагу 10.

7. Принять $n = n + 1$.

8. Если $i = j$, то перейти к шагу 9, иначе — к шагу 10.

9. Принять $m = m + 1$ и перейти к шагу 12.

10. Если $i \neq 1$, то перейти к шагу 11, иначе — к шагу 12.

11. Принять $i = i - 1$ и перейти к шагу 5.

12. Если $j \neq 1$, то перейти к шагу 13, иначе — к шагу 14.

13. Принять $j = j - 1$ и перейти к шагу 5.

14. Указать противнику числа m и n .

15. Если $m = 5$, то число угадано противником за k попыток, иначе перейти к шагу 2 для оценки следующего варианта противника.

Этот громоздкий алгоритм достаточно полно характеризует отличия между распознаванием даже простейших образов человеком и дискретным автоматом, обрабатывающим информацию отдельными порциями. Человеку достаточно бросить взгляд на сравниваемые числа для определения совпадающих цифр и их расположения. Действия же машины в этом случае подобны действиям слепого, на ощупь сравнивающего каждый предмет из некоторого их множества для сравнения с заданными образцами.

При таком сравнении на ощупь слепой попадает в затруднительное положение, когда требуется оценить число совпадений двух или нескольких одинаковых предметов в предъявляемом для оценки и эталонном множествах и число совпадений в

порядке сравнения элементов этих множеств. Неоднозначность в определении числа n цифр, когда числа a и b содержат по две или более одинаковых цифры, характерна и для составленного алгоритма.

В этом алгоритме сравнение цифр a_i и b_j прекращается при совпадении этих цифр в одинаковых разрядах. Если, например, цифре числа b соответствуют две одинаковые цифры числа a , находящиеся в разрядах с номерами $i \geq j$, то при первом совпадении цифр перебор прекратится и число n увеличится на единицу. Если же одинаковые цифры числа G находятся в разрядах с номерами $i \leq j$, то при первом совпадении цифр число n увеличится на единицу, но разряды не будут совпадать, и перебор цифр продолжится до второго совпадения разрядов с одинаковыми цифрами, при котором n увеличится еще на единицу. Следовательно, число n будет зависеть от расположения повторяющихся цифр в разрядах чисел a и b .

Этот недостаток подсчета n можно устраниТЬ, но тогда алгоритм усложнится, а время выполнения реализующей его программы возрастет. Поэтому сохраним алгоритм в составленном виде, учитывая, что основным критерием приближения к числу a является число m , завышенные значения числа n образуются относительно редко.

«Загадывание» микрокалькулятором неизвестного противнику числа a обеспечивается при его случайном выборе. Случайное целое число можно получить, например, с помощью вспомогательной мини-программы Сх П6 КИП6 БП А, где А — адрес оператора КИП6. После пуска такой программы образуется цикл с неограниченным числом итераций, на каждой из которых содержимое регистра 6 увеличится на единицу. При аварий-

ной остановке такой программы в случайный момент времени содержимое регистра 6 также будет случайным. Его преобразование в число a обеспечивается фрагментом ИП6 ln 1 ВП 5×П8 КИП8, после выполнения которого в регистре 8 содержится шестизначное целое число, пять последних цифр которого можно принять в качестве «загадываемого» числа a . Эти цифры будем хранить соответственно в регистрах 9, A, ..., D, используя для этого цикл с выделением нормированных цифр $a_i/10$ и их косвенной засылкой в соответствующие регистры памяти.

Для хранения изменяющихся на единицу целых чисел i, j, n, m и k выделим соответственно регистры 0, 1, 4, 5 и 6, используя их в качестве

адресных регистров-счетчиков в операторах косвенного вызова из памяти. Все операции по формированию «загадываемого» числа a и занесению в память вспомогательных коэффициентов $14 = P_2$ и $10^5 = P_3$ целесообразно реализовать фрагментами, следующими после основной части программы, что обеспечит ее выполнение при нажатии только клавиш С/П или В/О и С/П. Подобная реализация алгоритма представлена программой 5.

Для проверки введенной программы следует пустить ее нажатием клавиш БП 8 5 С/П и через несколько секунд остановить нажатием клавиш С/П или В/О, проверив содержимое регистра 6, которое должно быть положительным целым числом с нулями в

Программа 5. Компоновка чисел при «загадывании» микрокалькулятором пятизначного числа

П8 48	ИП3 63	+	10	П8 48	Сх 0Г	П4 44	П5 45	ИП2 62	П1 41	КИП6 Г6
КИП1 Г1	ИП2 62	П0 40	ИП8 68	1 01	0 00	÷ 13	П8 48	КИП8 Г8	→ 25	
ИП8 48	— 11	П7 47	ИП7 67	КИП0 Г0	— 11	х = 0 5Е	37 37	КИП4 Г4	ИП0 60	
ИП1 61	— 11	х = 0 5Е	37 37	КИП5 Г5	БП 51	42 42	ИП0 60	9 09	— 11	
х = 0 5Е	23 23	ИП1 61	9 09	— 11	х = 0 5Е	10 10	ИП4 64	ИП5 65	С/П 50	
БП 51	00 00	ИП6 66	ln 18	1 01	ВП 0С	5 05	П3 43	×	П8 48	
КИП8 Г8	1 01	4 04	П2 42	П0 40	ИП8 68	1 01	0 00	÷ 13	П8 48	
КИП8 Г8	→ 25	ИП8 68	— 11	КП0 L0	ИП0 60	9 09	— 11	х = 0 5Е	65 65	
П6 46	БП 51	49 49	Сх 0Г	П6 46	КИП6 Г6	БП 51	85 85			

Инструкция. Нажать клавиши БП 8 5 С/П, через 5–10 с нажать клавишу С/П для остановки программы, затем клавиши БП 5 2 С/П, что приведет к высвечиванию нуля; далее вводить вариант отгадываемого числа в регистр X , нажимая после каждого ввода кла-

вишу С/П или клавиши В/О и С/П и учитывая получаемые значения $PX = P_5 = n$ – числа разрядов с угаданными цифрами, $PY = P_4 = m$ – числа угаданных цифр и $P6 = k$ – числа попыток угадывания.

старших разрядах. Затем следует ввести число 10 в регистр 6 и нажать клавиши БП 5 2 С/П. После выполнения программы (время счета около 30 с) содержимое регистров $PX = P6 = 0, P2 = 14, P3 = -10^5, P9 = 0,3; PA = 0, PB = 0,2; PC = 0,5; PD = 0,8$, что соответствует задуманному числу $a = 30258$. Теперь при вводе в регистр X отгадываемых вариантов числа 10 000, 85 203 и 30 258 с нажатием после каждого ввода клавиши С/П должно получиться $PX = P5 = m = 1, PY = P4 = n = 4$ (время счета около 1 мин 50 с), $m = 1, n = 5$ (время счета около 2 мин) и $m = 5, n = 5$ (время счета около 1 мин 30 с).

Длину программы 5 можно сократить, вынося в подпрограмму одинаковые фрагменты подготовительной и основной частей программы, но в этом случае время счета еще больше увеличится.

Для участия в этой игре микро-ЭВМ в качестве отгадывающего партнера необходимо прежде всего выбрать наилучший путь (стратегию) отгадывания числа за минимальное число попыток. Человек в этом случае меняет стратегию в зависимости от получаемых значений m и n , но при программировании, например, микрокалькуляторов приходится ограничиться подсказкой только числа m разрядов с угаданными цифрами, так как учет числа цифр n громоздок и нереализуем при небольшом ресурсе памяти.

Простейший метод компоновки числа по значениям m заключается в последовательной подстановке в каждый из пяти разрядов отгадываемого числа b цифр от 0 до 9 с переходом к следующему разряду после увеличения числа m . Так как уже первый вариант числа может оказаться верным, а в наихудшем случае потребуется еще 45 попыток угадывания,

то среднее число таких попыток равно 23.

Перебор цифр с их увеличением, начиная с младших разрядов, аналогичен вращению колесиков автомобильного счетчика километров. Развличие заключается в том, что очередное колесико автомобильного счетчика начинает вращаться только после достижения цифры 9 в предыдущем разряде, тогда как наше «колесико» начинает вращаться сразу после угадывания содержимого предыдущего разряда. Подобный «счетчик» можно реализовать различными способами. Так как языки программирования лучше приспособлены для вычислений, то проще всего использовать арифметические операции. Например, выбрав для первой попытки число $b = 99\ 999$, при последующих попытках следует принимать $b = b - d$, где для самого младшего разряда $d = 1$, а для каждого следующего d увеличивается в 10 раз.

Необходимая микро-ЭВМ информация о степени приближения к загаданному числу содержитя в разности $\Delta t = m_k - m_{k-1}$ для двух очередных попыток и числе $m = 5$, свидетельствующем об «угадывании» содержимого всех пяти разрядов. Однако после первого «угадывания» $b = 99\ 999$ разность Δt оказывается неопределенной и действия микро-ЭВМ после этой и следующих попыток должны быть различными. Для выбора требуемых действий часто используют программные «узелки на память», называемые *флагами*; входные языки многих микро-ЭВМ содержат операторы установки, снятия и проверки флагов. При отсутствии таких операторов в качестве флага можно выбрать хранящуюся в памяти величину f , принимающую значения $f = 0$ (флаг установлен) или $f \neq 0$ (флаг не установлен), проверяемые условными операторами.

Если загаданное число a содержит разряды с цифрами 9, то при переходе к такому разряду и уменьшении его содержимого число m уменьшится и будет выполняться неравенство $\Delta m < 0$. В таких случаях следует восстановить цифру 9 и перейти к следующему разряду, последовательно выполнив операции присваивания $b = b + d$, $d = 10d$, $b = b - d$.

С учетом этих соображений составим следующий алгоритм (при использовании числа k попыток для формирования флага), реализующий выбранный метод компоновки многозначных чисел:

1. Принять число попыток $k = 0$, $d = 1$, $m = 0$, $b = 99\ 999$.

2. Запомнить число m разрядов с «угаданными» цифрами.

3. Если число m равно числу разрядов задуманного числа, то оно «угадано», иначе перейти к шагу 4.

4. Принять $k = k + 1$, $\Delta m = m_k - m_{k-1}$.

5. Если $k = 1$, то перейти к шагу 10, иначе — к шагу 6.

6. Если $\Delta m = 0$, то перейти к шагу 10, иначе — к шагу 7.

7. Если $\Delta m < 0$, то перейти к шагу 8, иначе — к шагу 9.

8. Принять $b = b + d$.
9. Принять $d = 10d$.
10. Принять $b = b - d$ и перейти к шагу 2.

На рассмотренном языке программирования микрокалькуляторов при размещении текущих данных $k = P5$, $\Delta m = P6$, $d = P7$, $b = P8$, $m = P9$ и загадывании пятизначного числа этот алгоритм реализуется программой 6. Для компоновки чисел с $n \leq 8$ разрядов достаточно заменить в этой программе оператор 5 по адресу 06 оператором набора числа n .

Если, например, загадано число 78 945, то микрокалькулятор «отгадает» его по программе 6 за 14 попыток. Эта программа отличается «игровым эффектом» лишь при небольшом числе партий с различными противниками. Партнер, сыгравший несколько партий, раскрывает нехитрую стратегию микрокалькулятора и ставит его в наиболее затруднительное положение, задумывая числа 0 или 10 000. Эту возможность можно исключить, формируя при первой попытке число b случайным образом с помощью ранее рассмотренных способов, но длина программы в этом случае увеличится.

Программа 6. Компоновка чисел при «отгадывании» микрокалькулятором пятизначного числа

ИП9 69	ХУ 14	П9 49	— 11	П6 46	ИП9 69	5 05	— 11	$x \neq 0$ 57	34 34
КИП5 Г5	ИП5 65	1 01	— 11	$x \neq 0$ 57	30 30	ИП6 66	$x \neq 0$ 57	30 30	$x \geq 0$ 59
25 25	ИП8 68	ИП7 67	+ 10	П8 48	ИП7 67	1 01	0 00	× 12	П7 47
ИП8 68	ИП7 67	— 11	П8 48	ИП8 68	С/П 50	БП 51	00 00	Сх ОГ	П5 45
П9 49	1 01	П7 47	9 09	9 09	9 09	9 09	9 09	БП 51	33 33

Инструкция. Нажать клавиши БП 3 8 С/П, что приведет к высвечиванию числа $b = 99\ 999$; после этого и последующих высвечиваний «угадываемых» чисел вводить в ре-

гистр X число m разрядов с «угаданными» цифрами и нажимать клавиши В/О и С/П или С/П; число попыток угадывания хранится в регистре 5.

Исходные условия некоторых игр с отгадыванием чисел обеспечивают выбор оптимальной стратегии отгадывания. В частности, это относится к отгадыванию целых чисел из интервала $(0; 2^n)$ с кратной двум верхней границей интервала. В этом случае для отгадывания задуманного числа от 1 до $2^n - 1$ наиболее эффективна стратегия, основанная на половинном делении интервала, содержащего искомое решение, до получения этого решения. В наихудшем случае искомое нечетное число определяют при последней попытке делением интервала шириной $\Delta = 2$ пополам. В этом случае ширина заключительного интервала $\Delta = 1 = 2^n / 2^k$ и максимальное число попыток $k = \ln 2^n / \ln 2 = n$. Например, при выборе верхней границы исходного интервала $2^{16} = 65\ 536$ любое число в этом интервале будет «угадано» методом половинного деления не более чем за 16 попыток.

В такой игре для заданного числа a из интервала $(0; 2^n)$ после каждого числа b , выбиравшегося отгадывающим, загадывающий партнер отвечает «увеличить», или «уменьшить» или «угадано». Затем партнеры меняются ролями, и после окончания второй партии выигрыш определяется по минимальному числу попыток угадывания. Алгоритм «отгадывания» методом половинного деления отображается следующим описанием:

1. Принять число попыток $k = 0$, отгадываемое число $b = 2^n / 2$.
 2. Принять $k = k + 1$.
 3. Если $b = a$, то число отгадано за k попыток, иначе перейти к шагу 4.
 4. Если $b < a$, то перейти к шагу 5, иначе — к шагу 6.
 5. Принять $b = b + b/2$ и перейти к шагу 2.
 6. Принять $b = b - b/2$ и перейти к шагу 2.
- Этот алгоритм несложно реализовать на любом языке программиро-

вания при отображении подсказок противника «увеличить» и «уменьшить» в виде сохранения варианта «угадываемого» числа b или, соответственно, изменения его знака на отрицательный. Программную реализацию такого алгоритма можно дополнить «загадывающей» частью с формированием «загадываемого» числа по случайному закону и подсказок отгадывающему это число противнику, реализуемых сравнением «загаданного» числа a и «отгадываемых» чисел b . Подобная реализация на входном языке микрокалькуляторов представлена программой 7.

Исходное число 2^n должно быть целым, и поэтому для его вычисления нельзя использовать оператор X^y . Пусть выбрано $2^{16} = 65\ 536$, а противник задумал число $a = 30\ 000$. Тогда при правильных подсказках противника микрокалькулятор последовательно высвечивает ответы $b = 32\ 768, 16\ 384, 24\ 576, 28\ 672, 30\ 720, 29\ 696, 30\ 208, 29\ 952, 30\ 080, 30\ 016, 29\ 984, 30\ 000$, «отгадав» число a за 12 попыток. Микрокалькулятор, «обученный» по программе 7, всегда «угадывает» задуманное число за минимальное число попыток, тогда как его противник, не знакомый с оптимальной стратегией игры, может лишь случайно угадать число за такое же число попыток.

Многие логические задачи связаны с перемещениями элементов множеств. Классическим примером является задача о ханойской башне или пирамидке [5] и ее разнообразные аналоги от шуточных [10] до прикладных [9]. В этой задаче заданы r дисков различного размера с отверстиями и три колышка A, B и C . В исходном состоянии (рис. 8) все диски, пронумерованные числами 1, 2, ..., r в порядке увеличения размеров дисков, надеты на колышек A .

Программа 7. «Загадывание» и «отгадывание» целых чисел из интервала $(0; 2^n)$

П8 48	КИП4 Г4	ИП5 65	2 02	\div	13	П5 45	ИП3 63	$x = 0$ 5E	23 23	ИП8 68
$x \geq 0$ 59	16 16	ИП5 65	+	10	БП 51	19 19	ИПС 6C	ИП5 65	— 11	ПС 4C
С/П 50	БП 51	00 00	ИП8 68	ИП7 67	—	11	$x \neq 0$ 57	20 20	$x < 0$ 5C	33 33
1 01	БП 51	20 20	1 01	/ — / 0L	БП 51	20 20	Сх 0Г	П3 43	П4 44	
ИП9 69	2 02	\div	13	П5 45	БП 51	19 19	ИП6 66	\sin 1C	x^2 22	$\sqrt{ }21$
ИП9 69	1 01	—	11	\times 12	П3 43	П7 47	КИП7 Г7	Сх 0Г	П4 44	БП 51
20 20	Сх 0Г	П6 46	КИП6 Г6	БП 51	63 63					

Инструкция. Установить переключатель Р — Г в положение Р, ввести $2^n = P9$; 1) при «загадывании» числа микрокалькулятором нажать клавиши БП 6 1 С/П и через 5—15 с остановить выполнение программы нажатием клавиши С/П или П3, затем нажать клавиши БП 4 6 С/П, что приведет к высвечиванию нуля; после каждого ввода противником в регистр X отгадываемого числа нажать клавиши В/О и С/П или С/П, что приведет к высвечиванию $PX = 1$ («увеличить»), $PX =$

= —1 («уменьшить») или $PX = 0$ («число отгадано»); 2) при «отгадывании» числа микрокалькулятором нажать клавиши БП 3 7 С/П, что приведет к высвечиванию $PX = 2^n/2$; после каждого выполнения программы и высвечивания «отгадываемого» числа противник нажимает клавиши В/О и С/П или С/П без изменения содержимого регистра X (если высвечиваемое число меньше «загаданного») или предварительно нажимает клавишу / — / (если высвечиваемое число больше «загаданного»); число попыток хранится в регистре 4.

Требуется за минимальное число ходов (перемещений одного из дисков на другой колышек так, чтобы большие диски не располагались над меньшими) переместить все диски на колышек С.

Оптимальное решение этой задачи [1], соответствующее минимальному числу $2^r - 1$ ходов, заключается в перемещении на каждом m -м ходу диска с номером, равным номеру первого справа разряда, содержащего единицу, в двоичном представлении числа r . При этом для четного числа r диск с четным или нечетным номером перемещается соответственно в последовательности колышков АСВА или АВСА, а для нечетного r — в последовательности АВСА или АСВА.

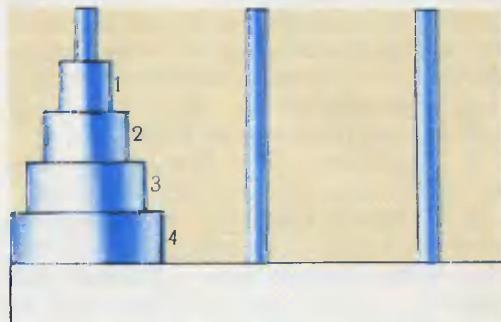


Рис. 8.

Так как двоичное представление нечетного номера хода содержит единицу в младшем разряде его двоичного представления, то на каждом нечетном ходу должен перемещаться наименьший диск с номером 1. Следовательно, при четном ходе возможно лишь перемещение меньшего из дисков на двух остальных колышках. Поэтому решение задачи сводится к последовательному выполнению двух логических операций.

1. При нечетном ходе переместить наименьший диск на следующий (в порядке $ABCAB\dots$ при четном числе дисков r и $ACBAC\dots$ при нечетном r) колышек.

2. При четном ходе переместить меньший (из оставшихся на остальных двух колышках) диск на другой из остальных колышков.

Обозначим номерами $i = 1, 2, 3$ соответственно колышки A, B, C при четном r и колышки A, C, B при нечетном r , буквой n_i — множество номеров дисков на i -м колышке, записанных в порядке уменьшения номеров дисков как многозначное число, а символами $E(\)$ и $D(\)$ — целую и дробную части числа, содержащегося в скобках. Тогда решение рассматриваемой задачи можно отобразить следующим алгоритмом:

1. Принять $i = 1$.
2. Принять $n_i = (n_i - 1)/10$.
3. Принять $T = E(n_i/10)$, $t = = 10D(n_i/10)$.
4. Принять $i = i + 1$, но при $i = 4$ принять $i = 1$.
5. Принять $n_i = 10n_i + 1$ (нечетный ход).
6. Принять $i = i + 1$, но при $i = 4$ принять $i = 1$.
7. Принять $K = E(n_i/10)$, $k = = 10D(n_i/10)$.
8. Если $t < k$, то перейти к шагу 9, иначе — к шагу 12.
9. Принять $n_i = 10n_i + t$.

10. Принять $i = i + 1$, но при $i = 4$ принять $i = 1$.

11. Принять $n_i = T$ и перейти к шагу 15.

12. Принять $n_i = K$.

13. Принять $i = i + 1$, но при $i = 4$ принять $i = 1$.

14. Принять $n_i = 10n_i + k$.

15. Принять $i = i + 1$, но при $i = 4$ принять $i = 1$ (четный ход).

16. Перейти к шагу 2.

Программная реализация этого алгоритма, обеспечивающего минимальные требования к емкости памяти, зависит от особенностей входного языка микро-ЭВМ. При реализации алгоритма на языках персональных ЭВМ, допускающих вывод на дисплей графической информации, его целесообразно дополнить операторами, обеспечивающими изображение хода решения задачи. При использовании программируемых микрокалькуляторов рассмотренных типов для хранения множеств номеров дисков на колышках целесообразно выбрать регистры памяти 7, 8 и 9, наиболее удобные для вызова в индикаторный регистр их содержимого. В этом случае описанный алгоритм реализуется программой 8, которая может быть сокращена при использовании микрокалькуляторов «Электроника МК-61» или «Электроника МК-52», входной язык которых содержит операторы для выделения целой и дробной частей числа.

При $r = 3$ и исходных данных $321 = PA, O = PB = PC$ после каждого выполнения этой программы получаются множества номеров дисков на соответствующих колышках $(A, C, B)_k = (32, 0, 1)_1, (3, 2, 1)_2, (3, 21, 0)_3, (0, 21, 3)_4, (1, 2, 3)_5, (1, 0, 32)_6, (0, 0, 321)_7$. Как и следовало ожидать, задача решена за минимальное число $(2^3 - 1 = 7)$ ходов (время каждого выполнения программы составляет около 30 с).

Программа 8. Решение задачи о пирамидке при $r \leq 8$

Сх 0Г	П8 48	П9 49	7 07	ПА 4—	1 01	0 00	ПВ 41	КИПА Г—	1 01
— 11	ИПВ 6L	÷ 13	КПА L—	ПП 53	72 72	ПС 4C	ИПО 60	П4 44	ПП 53
60 60	ИПВ 6L	× 12	1 01	+ 10	КПА L—	С/П 50	ПП 53	60 60	ПП 53
72 72	ПД 4Г	ИПС 6C	— 11	$x < 0$ 5C	49 49	ИПО 60	ПП 53	59 59	ИПВ 6L
× 12	ИПД 6Г	+ 10	КПА L—	С/П 50	ПП 53	60 60	БП 51	08 08	КИПА Г—
ИПВ 6L	× 12	ИПС 6C	+ 10	ПП 53	59 59	ИП4 64	БП 51	43 43	КИПА Л—
ИПА 6—	1 01	+ 10	ПА 4—	ИПВ 6L	— 11	$x = 0$ 5E	70 70	7 07	ПА 4—
КИПА Г—	В/О 52	ИПВ 6L	÷ 13	П5 45	1 01	+ 10	П0 40	КИПО Г0	ИП5 65
ИПО 60	— 11	ИПВ 6L	× 12	$x = 0$ 5E	87 87	ИПВ 6L	В/О 52		

Инструкция. Ввести в регистр 7 номера дисков $r, \dots, 2, 1$ в порядке их уменьшения; для первого хода нажать клавиши В/О и С/П, для последующих — только клавишу С/П;

Реализация рассмотренного алгоритма на входных языках персональных ЭВМ, обеспечивающих вывод графической информации, значительно более эффективна при показе на дисплее изменяющегося изображения дисков на колышках после каждого хода.

По условиям задачи о пирамидке множества номеров дисков упорядочены в соответствии с размерами. Однако во многих логических задачах приходится выполнять операции над неупорядоченными множествами элементов. Рассмотрим, например, следующую задачу о формировании железнодорожного состава: из восьми или менее вагонов с произвольной нумерацией, находящихся на участке B (рис. 9), требуется сформировать на участке A состав с обратным

после каждого выполнения программы в регистрах 7, 8 и 9 хранятся номера дисков соответственно на колышках A , B и C при четном числе r дисков или A , C и B при нечетном r .

порядком следования вагонов при использовании тупиков C , вмещающего один вагон, и D , вмещающего весь состав. Посредством логических умозаключений составим следующий алгоритм оптимального (за минимальное количество операций) решения:

1. Отделить на участке B один вагон справа.
2. Если отделенный вагон единственный на участке B , то перейти к шагу 3, иначе — к шагу 4.
3. Присоединить вагон к составу на участке A и закончить решение.
4. Переместить отданный вагон в тупик C .
5. Переместить вагоны с участка B в тупик D .
6. Присоединить вагон из тупика C к составу на участке A .
7. Переместить вагоны из тупика D

на участок B и перейти к шагу 1. Регистрируя находящиеся на каждом из участков пути вагоны после каждого их перемещения, получим с помощью такого алгоритма последовательно положения вагонов, соответствующие оптимальному решению рассматриваемой задачи о переформировании железнодорожного состава.

При реализации алгоритма на входном языке микрокалькуляторов для выделения элемента множества и его присоединения к другому множеству достаточно воспользоваться ранее описанными фрагментами. В этом случае при хранении множеств номеров вагонов, находящихся на участках A , B , C и D , в регистрах памяти с аналогичными обозначениями несложно реализовать алгоритм программой 9.

Для последовательности вагонов с номерами 2, 7 и 4, находящихся на участке B пути, по этой программе получим перемещения $(A, B, C, D)_k = = (0, 274, 0, 0)_0, (0, 27, 4, 0)_1, (0, 0, 4, 27)_2, (4, 0, 0, 27)_3 = (4, 27, 0, 0)_4 = = (4, 2, 7, 0)_5, (4, 0, 7, 2)_6 = (47, 0, 0, 2)_7 = (47, 2, 0, 0)_8 = (472, 0, 0, 0)_9$. Если предпоследнее размещение, соот-

ветствующее эксплуатационной проверке последнего вагона на участке B , не нужно, то можно дополнить программу фрагментом, соединяющим один элемент множества D с множеством A без промежуточного соединения с множеством B .

Решения приведенных задач связаны с отделением и присоединением по одному элементу множеств. Между тем многие задачи связаны с перебором всех элементов множеств для их сравнения с элементами других множеств. Напомним, что результаты логических операций над двумя множествами A и B называются объединением $C = A \cup B$, элементы которого принадлежат множеству A и (или) множеству B ; пересечением $C = A \cap B$, элементы которого принадлежат как множеству A , так и множеству B , и логической разностью $C = A \setminus B$, элементы которой принадлежат множеству A , но не принадлежат множеству B . Следовательно, в алгоритмах результат присоединения элемента к множеству A можно обозначить символом $A = A \cup a$, а выделение элемента a из множества A формулой $A = A \setminus a$.

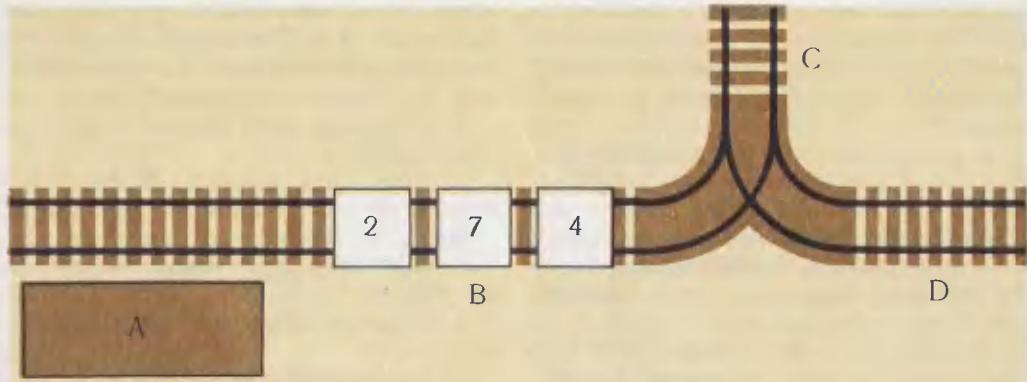


Рис. 9.

Программа 9. Решение задачи о формировании железнодорожного состава

ИПВ 6L	$x = 0$ 5E	07 07	ИПА 6—	С/П 50	БП 51	00 00	1 01	0 00	\div	13
П9 49	1 01	+ 10	П1 41	КИП1 Г1	ИП9 69	ИП1 41	— 11	1 01	0 00	
×	12	ИП 4C	ИП1 61	$x = 0$ 5E	30 30	ПП 53	49 49	ПВ 4L	БП 51	03 03
ПВ 4L	ИПА 6—	С/П 50	ИПВ 6L	ПД 4Г	Сх 0Г	ПВ 4L	ИПА 6—	С/П 50	ПП 53	
49 49	ИПА 6—	С/П 50	ИПД 6Г	ПВ 4L	Сх 0Г	ПД 4Г	БП 51	03 03	ИПС 6C	
ИПА 6—	1 01	0 00	×	12	+ 10	ПА 4—	Сх 0Г	ПС 4C	В/О 52	

Инструкция. Ввести множество номеров (от 1 до 9) восемьми или менее вагонов, находящихся на участке B , в регистр B и очистить регистры A , C и D ; после каждого выполнения программы (для первого пуска

С учетом этих обозначений рассмотрим популярную на заре развития кибернетики задачу о поиске выхода из лабиринта, реализуемую самообучающимися (адаптивными) программами работы ЭВМ. Предположим, что лабиринт состоит из нескольких помещений, соединенных проходами, один из которых ведет к выходу из лабиринта. Человек, очутившийся в одном из помещений такого лабиринта, чтобы избежать бесполезных блужданий, должен каким-либо способом отмечать проходы. Для выхода из исходного помещения он случайно выбирает проход и отмечает его. В следующих помещениях этот человек должен по определенной системе (например, по часовой стрелке) исследовать проходы из этих помещений, пропуская отмеченные ранее как тупиковые или пройденные. Если в очередном помещении имеется только один проход, то человек возвращается через него в предыдущее помещение, отметив этот проход как тупиковый. Если в очередном помещении все проходы оказались от-

нажать клавиши B/O и C/P , для последующих — только клавишу C/P) множества номеров A , B , C и D вагонов, находящихся на соответствующих участках размещены в регистрах $PX = PA = A$, $PB = B$, $PC = C$, $PD = D$.

меченными как тупиковые или пройденные, то человек также должен вернуться через проход, которым он впервые попал в это помещение и который он также отмечает как тупиковый. Исследуя таким образом встречающиеся на пути помещения, человек находит выход из лабиринта. При повторном попадании в лабиринт человек, минуя тупиковые проходы, находит более короткий путь.

Обозначая (или сохраняя предыдущие обозначения) проходы в новом помещении при их проверке по часовей стрелке последовательностью символов a_i , последним символом следует указывать проход, через который человек впервые попал в это помещение. Обозначив этот проход символом a_k , выход из лабиринта — символом b , а множества тупиковых и пройденных проходов — символами T и P , формализуем выбор прохода из очередного помещения следующим алгоритмом:

1. Принять $i = 1$.
2. Если $a_i = b$, то выход найден, иначе перейти к шагу 3.

3. Если $a_i = a_k$, то перейти к шагу 4, иначе — к шагу 5.

4. Принять $T = T \cup a_k$ и выйти в предыдущее помещение, перейдя для него к шагу 1 алгоритма.

5. Если $a_i \in T$ (a_i принадлежит множеству T), то перейти к шагу 7, иначе — к шагу 6.

6. Если $a_i \in \Pi$, то перейти к шагу 7, иначе — к шагу 8.

7. Принять $i = i + 1$ и выйти к шагу 2.

8. Принять $\Pi = \Pi \cup a_i$ и перейти в следующее помещение, приняв для него $a_k = a_i$ и перейдя к шагу 1 алгоритма.

Последовательное выполнение этого алгоритма для очередных помещений приводит к выходу из лабиринта, причем действия, требуемые алгоритмом от человека, могут успешно имитироваться ЭВМ, работающей по соответствующей программе. Так как при повторных поисках выхода из лабиринта сохранение множества T обеспечивает исключение ряда ненужных операций, то такие программы называют самообучающимися или адаптивными (приспособливающимися).

Формально решение рассматриваемой задачи основано на операциях объединения и логического вычитания множеств. Пусть, например, в процессе поиска выхода проходы лабиринта обозначены номерами, показанными на рис. 10, а. Тогда структура лабиринта отобразится совокупностью множеств номеров проходов из каждого помещения (1, 3), (1, 2, 4), (2, 5), (3, 6), (4, 7), (5), (6), (7, 8). При вычитании тупиковых множеств (5) и (6), соответствующих помещениям с одним выходом, получим множества (1), (1, 2, 4), (2), (3), (4, 7), (7, 8) с тупиковыми множествами (2) и (3). В результате вычитания этих множеств (1), (1, 4), (4, 7), (7, 8) множество (1) не является тупиковым, так как проход 1 является исходным. Объединение этих множеств (1, 4, 7, 8) и определяет множество проходов, ведущих из исходного помещения к выходу из лабиринта.

При выполнении операций проверки отношений $a_i \in T$ и $a_i \in \Pi$ в алгоритме требуется сравнивать элемент a_i со всеми элементами множеств T и Π , выделяемыми из этих множеств. Перебор элементов множеств в случае

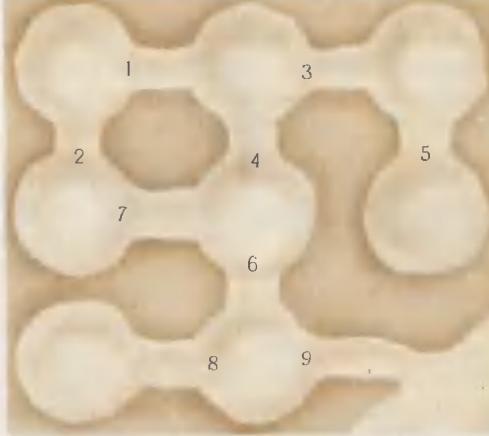
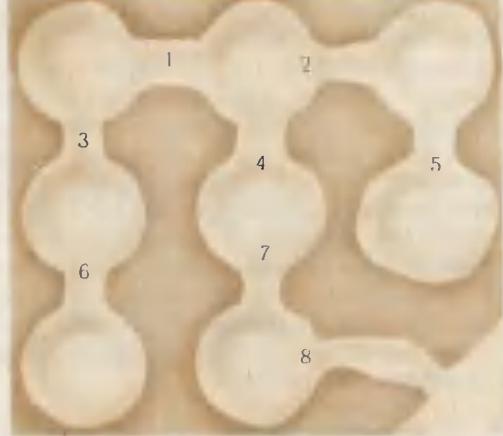


Рис. 10.

реализации алгоритма на входном языке программируемых микрокалькуляторов обеспечивается при $N_2 > 6$ фрагментом вида ИП₁ ПН₂ КИП₂ → ИП₂ — 1 0 × ПН₃, после выполнения которого в регистре N_1 хранится исходное нормализованное отделением запятой первого элемента множество a_1, a_2, \dots, a_k , в регистре N_2 — элемент a_1 этого множества, в регистре N_3 — нормализованная логическая разность a_2, a_3, \dots, a_k исходного множества и выделенного элемента. Последующее занесение содержимого регистра N_3 в регистр N_1 в цикле обеспечивает выделение всех элементов исходного множества.

При поиске микро-ЭВМ выхода из лабиринта пользователю приходится выполнять функции «органов чувств», вводя перед каждым выполнением программ для выбора прохода в очередное помещение множество номеров проходов из данного помещения. Полное число проходов в лабиринте ограничено разрядностью мантиссы программируемого микрокалькулятора, и при его использовании следует нумеровать проходы цифрами от 1 до 8 (в произвольном порядке), обозначая цифрой 9 проход, являющийся выходом из лабиринта (рис. 10, б). Для упрощения программы множество номеров проходов из данного помещения целесообразно вводить в нормализованном виде с запятой после первого номера, записывая последним номер прохода, через который данное помещение посещалось впервые. С учетом этих соображений, выделив регистры 7, С и Д для хранения соответственно нормализованного множества проходов, тупикового множества Т и множества П пройденных проходов, реализуем рассмотренный алгоритм программой 10. Для полной имитации действий человека, попавшего в лабиринт, первый номер в множестве

номеров выходов из исходного помещения следует выбрать случайным образом, не руководствуясь планом лабиринта, а перечисление номеров проходов целесообразно производить по часовой стрелке, начиная с того, через который посещено помещение первый раз.

Результаты выполнения этой программы точно имитируют действия человека, ищущего выход из лабиринта согласно принятым правилам. Так, предположим, что выходы из исходного помещения человек обозначил символами 1 и 2 и случайно выбрал выход 1. Попав в следующее помещение с тремя выходами, он отмечает их, согласно принятым правилам, по часовой стрелке от исходного прохода, через который он впервые вошел в помещение, например, номерами 3, 4 и 1 (выбор номеров произволен, но их последовательность должна отвечать правилам) и, естественно, выбирает проход 3 в следующее помещение. Пройдя в следующее помещение с двумя входами, отмечаемыми как 5 и 3, он попадает, например, в следующее помещение с одним входом и вынужден вернуться в помещение с входами 3, 4 и 1, отметив проходы 3 и 5 как тупиковые. Выбрав (при отсчете по часовой стрелке от исходного прохода 1) проход 4, он перейдет в следующее помещение.

Так как ввод пользователем множества номеров очередного помещения перед каждым пуском программы выполняется по тем же правилам, которыми руководствуется попавший в лабиринт человек, то его действия не подменяются пользователем. Если человек попал в помещение с двумя входами лабиринта, план которого (не известный человеку, попавшему в лабиринт) показан на рис. 10, б, то действия микрокалькулятора отображаются данными табл. 2. После первого поиска выхода из лабиринта

2. Поиск выхода из лабиринта по программе 10

Первый поиск			Повторный поиск ($PC = 35$)		
Ввод	Время счета, с	Выбор	Ввод	Время счета, с	Выбор
1,2	15	1	1,2	15	1
3,41	18	3	3,41	30	4
5,3	22	5	6,74	20	6
5	9	5	9,86	4	0
5,3	15	3			
3,41	35	4			
6,74	30	6			
9,86	4	0			

тупиковое множество содержит $PC = 35$, а множество пройденных проходов $PД = 6453$. Если сохранить содержимое регистра C , очистив регистр D , то (при условии, что случайно снова будет выбран проход 1 или человек сознательно его выберет с учетом приобретенного опыта) при повторном поиске, как видно из данных табл. 2, выход будет найден быстрее.

Программа 10. Поиск с самообучением выхода из лабиринта

ИП7 47	КИП7 Г7	→ 25	ИП7 67	— 11	П4 44	ИП7 67	9 09	— 11	$x \neq 0$ 57
75 75	ИП4 64	$x \neq 0$ 57	67 67	ИПС 6C	П8 48	КИП8 Г8	→ 25	ИП8 68	— 11
ИП7 67	ИП8 68	— 11	$x \neq 0$ 57	52 52	→ 25	$x \neq 0$ 57	33 33	1 01	0 00
×	БП 51	15 15	ИПД 6Г	П9 49	КИП9 Г9	→ 25	ИП9 69	— 11	ИП7 67
ИП9 69	— 11	$x \neq 0$ 57	52 52	→ 25	$x \neq 0$ 57	58 58	1 01	0 00	× 12
БП 51	34 34	ИП4 64	1 01	0 00	×	БП 51	00 00	ИПД 6Г	1 01
0 00	÷ 13	ИП7 67	+ 10	ПД 4Г	БП 51	74 74	ИПС 6C	1 01	0 00
÷ 13	ИП7 67	+ 10	ПС 4C	ИП7 67	С/П 50	БП 51	00 00		

Инструкция. Очистить регистры C и D , перед каждым пуском программы вводить в регистр X нормализованное (с запятой после первого номера) множество номеров проходов из очередного помещения (с последним номером прохода, через который помещение было посещено впервые, и обозначением выхода номером 9) и нажимать клавиши В/О и С/П

(для первого пуска программы обязательно) или С/П; после выполнения программы выдается выбираемый номер прохода в следующее помещение или нуль при выходе из лабиринта (множество номеров пройденных проходов хранится в регистре D); при повторном поиске очистить регистр D , сохранив содержимое регистра C .

Глава

4

ИГРЫ С ПОЛНОЙ ИНФОРМАЦИЕЙ

Различные игры с двумя* или более участниками, по очереди принимающими решения (называемые *ходами*) об изменении игровой ситуации, относятся к одной из двух основных групп. Первая из них включает игры с полной информацией, участник которых принимает очередное решение после того, как становится известным предыдущий ход противника.** Теоретически для таких игр возможен перебор всех вариантов и выбор из них наилучшего варианта, обеспечивающего выигрыш или, по крайней мере, минимальный проигрыш одного из участников, которым может быть и ЭВМ.

Эта особенность игр с полной информацией наглядно описана в старом кибернетическом анекдоте об игре двух ЭВМ в шахматы: после первого хода одной из них вторая, перебрав все варианты, отвечает: «Сдаюсь!». В этой истории анекдотично лишь предположение о возможности перебора всех вариантов шахматной игры. Полное их число порядка 2^{116} превышает число электронов во Вселенной и, следовательно, в нашем мире не хватит материала для создания ЭВМ, играющей в шахматы подобным методом. Вместе с этим описанная в анекдоте ситуация характерна для игр с небольшим числом вариантов,

которые возможно перебрать для выбора наилучшего, обеспечивающего успешное участие ЭВМ в игре. В качестве примера рассмотрим известную шахматную задачу Лойда, в которой белые начинают игру и ставят мат черному королю за три хода при исходном положении фигур, показанном на рис. 11. Оптимальный алгоритм решения этой задачи в шахматной записи имеет вид: 1. d4 Kph5 2. Фd3 Kph4 (g4) 3. Фh3 × или 1... Kpg4 2. e4 + Kph4 3. g3x[4].

Реализацию этого алгоритма можно поручить даже программируемому микрокалькулятору, но следует предварительно условиться о цифровом кодировании ходов. Для этого поля шахматной доски, занимаемые на первом ходу пешками, будем обозначать цифрами xy , где x и y — номера полей по горизонтали и вертикали. Для других фигур в общем случае целесообразно использовать трехзначный код zxy , где x и y определяют поле, занимаемое фигурой с номером (считая слева направо) z в исходном положении фигур.

При использовании подобного кодирования ходов решение рассмотренной трехходовой задачи на входном языке микрокалькуляторов обеспечивается программой 11, составление которой не требует пояснений.

Подобным образом по шахматной записи оптимального варианта игры можно программировать решение не только шахматных этюдов, но и не очень сложных вариантов игры в шахматы. К ним, в частности, относится

* В дальнейшем участника, начинающего игру, для краткости будем называть первым, а его противника — вторым участником.

** Ко второй группе относятся игры с неполной информацией (см. гл. 5).

Программа 11. Решение шахматной задачи Лойда

Cx 0Г	4 04	4 04	С/П 50	↑ 0Е	8 08	5 05	— 11	x = 0 5E	21 21
4 04	4 04	3 03	С/П 50	↑ 0Е	4 04	8 08	3 03	↑ 0Е	0 00
С/П 50	↑ 0Е	5 05	4 04	С/П 50	↑ 0Е	7 07	3 03	↑ 0Е	0 00
С/П 50	БП 51	00 00							

Инструкция. Нажать клавиши В/О и С/П (для последующих партий можно нажимать только клавишу С/П) для высвечивания первого хода белых 44 (или d4), затем вводить код *xy* поля, занимаемого королем

старинная игра «магараджа», где белые играют по обычным правилам, но достигшая края доски пешка не заменяется другой фигурой. Черные обладают лишь одной фигурой, называемой магараджей и обладающей правами ферзя и коня одновременно (рис. 12).

Наиболее короткий вариант победы белых над магараджей описан в книге [4], но он не учитывает возможность произвольного расположения магараджи перед началом игры. Из-

при ходе черных и нажимать только клавишу С/П для высвечивания следующего хода белых или (при мате) нуля с хранением последнего хода белых в регистре Y.

менив соответственно последовательность ходов, составим следующую запись игры белых: 1. Кс3 2. а4 3. Ла3 4. Лb3 5. Kf3 6. h4 7. Лh3 8. Лg3 9. d4 10. Фd3 11. Фe4 12. Фd5 13. Лb7 14. Лb8. Если магараджа занимает поле a6, то 15. e4 16. Сg5, иначе 15. Сg5 16. e4.

Следует отметить, что боевые качества магараджи обеспечивают ему серьезные преимущества перед недостаточно опытным противником. Однако при игре с микрокалькуля-

Программа 12. Шахматная игра «магараджа»

Cx 0Г	1 01	2 02	П0 40	ИП0 60	1 01	— 11	x ≠ 0 57	13 13	КИП0 Г0
С/П 50	БП 51	04 04	4 04	4 04	5 05	С/П 50	↑ 0Е	1 01	2 02
7 07	С/П 50	↑ 0Е	8 08	7 07	8 08	С/П 50	↑ 0Е	1 01	6 06
— 11	x = 0 5E	38 38	ИПС 6C	С/П 50	ИПД 6Г	0 00	С/П 50	ИПД 6Г	С/П 50
ИПС 6C	0 00	С/П 50							

Инструкция. (454 = P1; 443 = P2; 44 = P3; 873 = P4; 883 = P5; 84 = P6; 763 = =P7; 123 = P8; 113 = P9; 14 = PA; 233 = =PB; 54 = PC; 375 = PD) нажать клавиши В/О и С/П для высвечивания первого хода белых PX = 233, или Кс3, далее вводить в

регистр X код *xy* поля, занимаемого магараджей и нажимать только клавишу С/П, что приведет к высвечиванию хода белых или (при последнем ходе белых) нуля с хранением последнего хода белых в регистре Y.

тором, «обученным» по реализующей приведенный вариант программе 12, противник может отвечать «Сдаюсь!» после первого хода микрокалькулятора, играющего белыми.

Таким образом, если универсальным ЭВМ высокой производительности до сих пор не удается обыгрывать в шахматы гроссмейстеров, то микрокалькулятор с программой 12

становится чемпионом мира по шахматной игре под названием «магараджа».

Подобным образом после перебора вариантов и выбора оптимального программируется участие микро-ЭВМ и в других простейших играх на шахматной доске, подобных «минишахматам» [2] или «мини-шашкам» на доске с уменьшенным числом

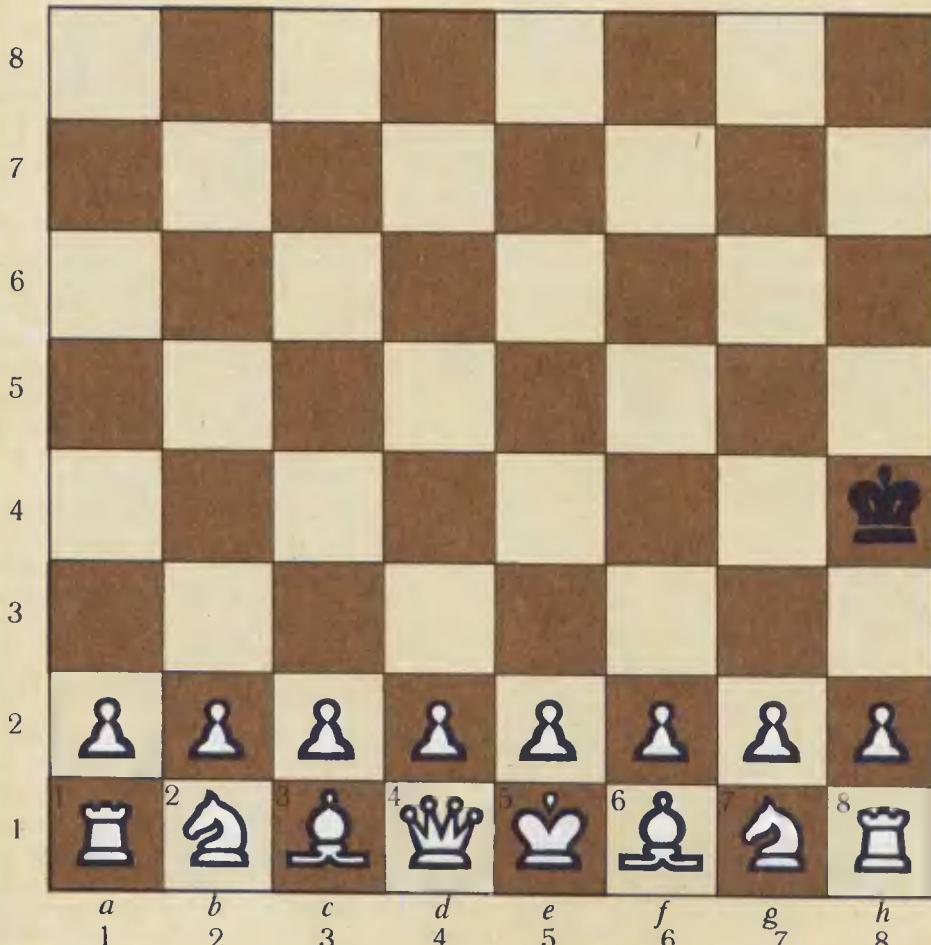


Рис. 11.

клеток. Метод полного перебора вариантов пригоден и для других игр, причем для выбора оптимального варианта обычно составляют граф * (дерево) игры с отображением возможных ходов участников игры. Приме-

* Графом называют чертеж, на котором линиями (ветвями графа) отображены связи между объектами, обозначенными условными символами (вершинами графа).

ром может служить игра Гранди, участники которой по очереди разбивают на две неравные части исходное множество из S предметов (например, кучку спичек), затем одно из полученных множеств и т. д. Выигрывает участник игры, после хода которого исходное множество разбито на множества, содержащие только один или два предмета.

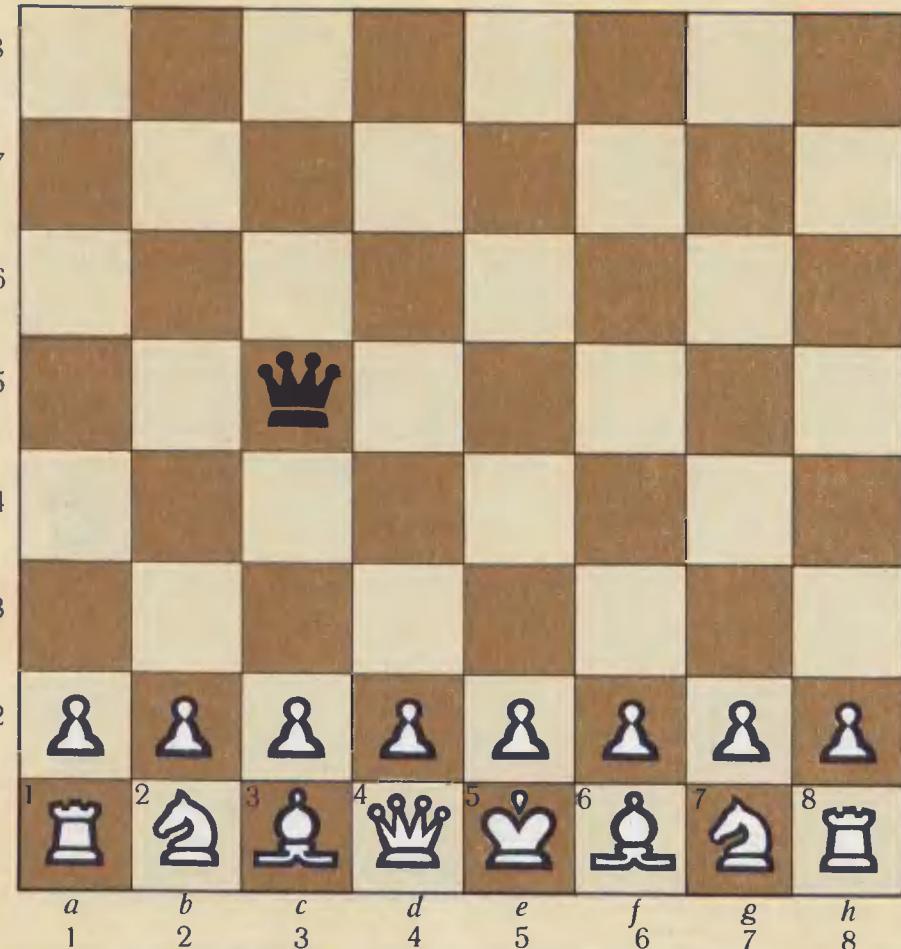


Рис. 12.

Программа 13. Игра Гранди при $S = 10$

П9 49	8 08	2 02	— 11	$x = 0$ 5Е	11 11	7 07	2 02	1 01	БП 51
14 14	6 06	3 03	1 01	C/П 50	П9 49	5 05	3 03	1 01	1 01
— 11	$x = 0$ 5Е	30 30	3 03	3 03	2 02	2 02	1 01	БП 51	35 35
4 04	2 02	2 02	1 01	1 01	C/П 50	Сх 0Г	2 02	2 02	2 02
1 01	1 01	1 01	1 01	↑ 0Е	0 00	C/П 50	БП 51	00 00	

Инструкция. Ввести в регистр X последовательность чисел 91, 82, 73 или 64, обозначающие числа предметов в множествах, на которые противник микрокалькулятора разбивает исходное множество с 10 предметами, и нажать клавиши В/О и С/П, что приведет к высвечиванию ответного хода микрокальку-

По графу такой игры видно, что, например, для $S = 6$ (рис. 13) первый участник выигрывает при разбиении исходного множества на два множества, содержащие по четыре и два предмета, но может проиграть при разбиении исходного множества на два по пять и одному предмету. Следовательно, оптимальному варианту игры соответствует начинаяющийся первым из этих двух возможных ходов.

Предельное число S предметов при участии в этой игре микро-ЭВМ определяется лишь ее характеристиками. При программировании микрокалькуляторов с отображением предметов в очередных множествах последовательностью однозначных чисел, высвечиваемых на индикаторе, максимальное число $S = 10$. Оптимальная игра микрокалькулятора в этом случае обеспечивается программой 13.

Если противник первым ходом разбивает исходное множество на два, например, с числами 8 и 2 предметов, т. е. вводит $82 = PX$, то микрокалькулятор отвечает ходом $PX =$

лятора, играющего вторым; далее аналогично вводить множества однозначных чисел (в порядке их уменьшения) предметов в множествах, включая разбитые, но нажимать только клавишу С/П; при выигрыше микрокалькулятора высвечивается нуль, а победное разбиение множества хранится в регистре Y .

$= 721$, разбивая первое множество на два с числами 7 и 1 предметом, при ходе противника $5221 = PX$ микрокалькулятор отвечает ходом $PX = 42\ 211$ и после единственного возможного хода противника $322\ 111 = PX$ выигрывает ходом $PX = 2\ 221\ 111$, высвечивая 0.

При увеличении исходного числа предметов сложность графов по-

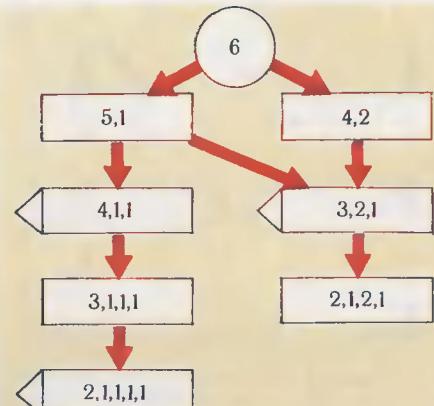
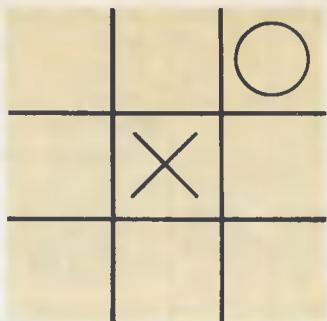
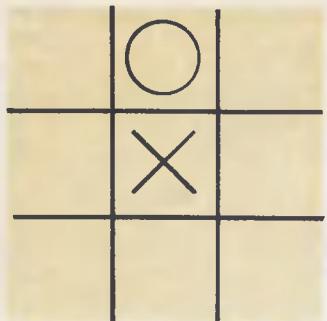


Рис. 13.



1	2	3
8	9	4
7	6	5

Рис. 14.

a

b

в

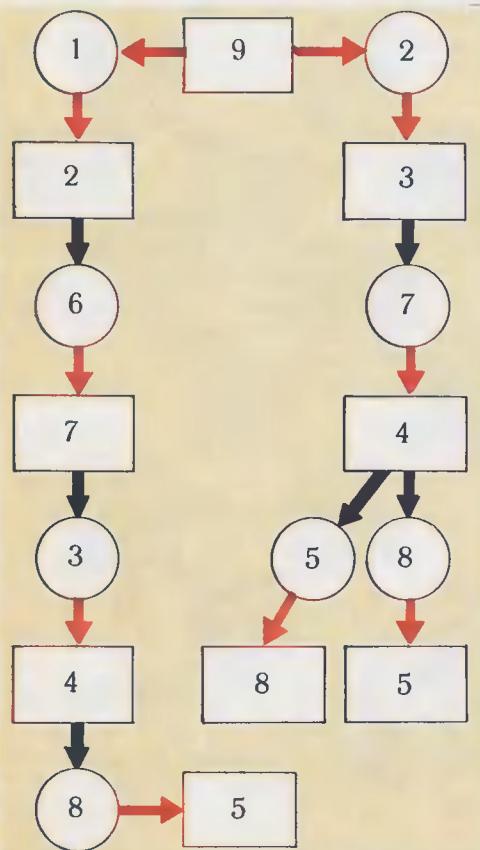


Рис. 15.

добрьих игр быстро возрастает, и составителю их оптимальных алгоритмов для игры машины приходится прибегать к различным приемам, обеспечивающим уменьшение числа перебираемых вариантов. Для многих игр это оказывается возможным при учете симметрии игровых ситуаций.

Примером может служить распространенная игра «крестики — нолики», участники которой по очереди отмечают крестиками и кружочками занимаемые клетки прямоугольной таблицы размера 3×3 , причем выигрывает занявший первым три смежных клетки по диагонали, вертикали или горизонтали. При программировании микро-ЭВМ с малым ресурсом памяти целесообразно представить ей первый ход для занятия центральной клетки таблицы. В этом случае возможны два варианта игры, исход которых при безошибочной игре обоих участников определяется первым ходом противника машины. Если он занимает не угловую клетку (четную при нумерации клеток, показанной на рис. 14, в), то игра заканчивается выигрышем микро-ЭВМ, при занятии же угловой (нечетной) клетки игра сводится к ничейному результату.

С учетом симметрии таблицы относительно ее клеток с четными и нечетными номерами составим граф игры (рис. 15) при занятии противником микро-ЭВМ своим первым ходом клетки с номером 2 (рис. 14, а) и клетки с номером 1 (рис. 14, б). В этих случаях ходы микро-ЭВМ следует выбирать так, чтобы противник был вынужден делать ответные ходы (показанные на рис. 15 черными стрелками) под угрозой немедленного проигрыша. При выборе противником ходов, отличающихся от указанных на графе, ходы микро-ЭВМ должны соответствовать номерам клеток, отличающимся от указанных на графике на такую же величину. Следовательно, достаточно выбрать стратегию игры микро-ЭВМ в зависимости от четности клетки, занимаемой первым ходом противника или ответным ходом машины. При этом в алгоритме игры должен быть предусмотрен выбор номеров клеток в тех случаях, когда вычисляемый

номер меньше 1 или больше 8. Этим требованиям отвечает следующее описание алгоритма, где символами m_i и n_i обозначены ходы микро-ЭВМ и ее противника:

1. Принять $m_1 = 9$.
2. Принять $x = n_1$ и выполнить шаги 9—11.
3. Если m_2 нечетно, то перейти к шагу 4, иначе — к шагу 6.
4. Принять $x = y$ и выполнить шаги 9—11.
5. Принять $m_4 = y$ и высветить 77 (выигрыш).
6. Принять $x = n_2$ и выполнить шаги 9—11.
7. Принять $x = n_4$ и выполнить шаги 9—11.
8. Высветить 0 (ничья).
9. Если $x - 1 = 0$, то принять $m_i = 8$, иначе принять $m_i = x - 1$.
10. Запомнить n_i ; если $m_i - 4 \leq 0$, то принять $y = m_i - 4 + 8$, иначе принять $y = m_i - 4$.
11. Если $y \neq n_i$, то принять $m_{i+1} = y$ и высветить 77 (выигрыш).

Программа 14. Игра «крестики — нолики»

↑ OE	9 09	C/P 50	P/P 53	25 25	π 20	× 12	cos 1Г	$x < 0$ 5C	17 17
ИП9 69	P/P 53	25 25	1 01	— 11	БП 51	48 48	ИП7 67	P/P 53	25 25
ИП7 67	P/P 53	25 25	0 00	C/P 50	1 01	— 11	$x = 0$ 5E	30 30	8 08
П9 49	C/P 50	P7 47	ИП9 69	4 04	— 11	$x \neq 0$ 57	40 40	$x < 0$ 5C	42 42
8 08	+	10	P8 48	ИП7 67	— 11	$x \neq 0$ 57	51 51	ИП8 68	7 07
C/P 50	ИП9 69	B/O 52							

Инструкция. Установить переключатель Р — Г в положение Р, нажать клавиши В/О и С/П для высвечивания номера $RX = 9$ центральной клетки, «занимаемой» микрокалькулятором; после каждого высвечивания номера клетки, «занимаемой» микрокалькулятором, вводить в регистр X номер клетки,

«занимаемой» противником, и нажимать только клавишу С/П; при выигрыше микрокалькулятора высвечиваются цифры 77 (номер клетки, «занимаемой» последним ходом микрокалькулятора, хранится в регистре Y); при ничейном результате высвечивается нуль.

На входном языке микрокалькуляторов этот алгоритм реализован программой 14, где шаги 9—11 представлены подпрограммой, первая часть которой реализует шаг 9 алгоритма с высвечиванием хода микрокалькулятора, а вторая часть — шаги 10 и 11.

Микрокалькулятор, «обученный» по программе 14, выигрывает при первом ходе противника на клетку с четным номером или сводит игру к ничейному результату в противном случае. Пример партии с микрокалькулятором: $B/O\ C/P\ PX=9$; $6=PX\ C/P\ PX=7$; $3=PX\ C/P\ PX=8$; $1=PX\ C/P\ PX=77$; $PY=4$.

Для определения четности чисел в программе 14 не использован фрагмент ИП8 $2-x \neq 0 A_1, x < 0 A_2$, где A_1 — адрес перехода при четном содержании регистра 8, а A_2 — адрес перехода к оператору 2 этого же фрагмента. Такой фрагмент на один шаг короче фрагмента, используемого для определения четности в предыдущих программах, но выполняется дольше и целесообразен лишь при небольших числах, содержащихся в регистре 8.

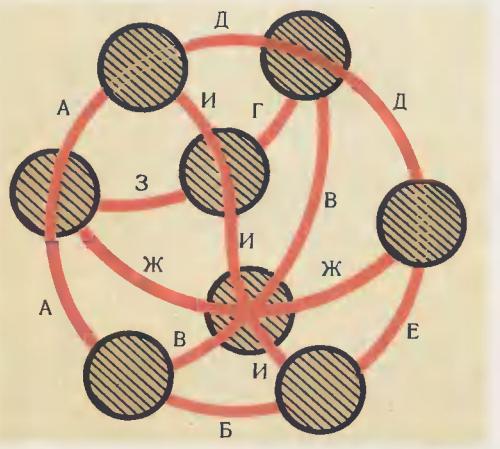


Рис. 16.

Игра «крестики-нолики», как и многие другие игры, имеет много аналогов, отличающихся по внешнему оформлению. Участники одной из таких игр [3] по очереди выбирают по одной из карточек со словами СОК, КЛИН, РЕКА, РЫБА, ТОР, НИТЬ, СЕТЬ, БУСЫ, НЕБО, причем выигрывает участник, первым набравший три карточки с одинаковой буквой. Если пронумеровать карточки цифрами от 1 до 9 в приведенной последовательности слов, то в этой игре также может успешно участвовать микрокалькулятор с программой 14.

Участники игры «дорожная пробка» [3] по очереди закрашивают карандашом определенного цвета одну из дорог карты, показанной на рис. 16. Выигрывает участник, первым закрасивший своим цветом все дороги, ведущие к одному из городов. Если заменить буквы, обозначающие дороги, в алфавитном порядке цифрами от 1 до 9, то одним из двух участников такой игры также может стать микрокалькулятор с введенной в память программой 14.

Участники еще одной игры по очереди выбирают игральные карты от

2	9	4
7	5	3
6	1	8

Рис. 17.

туза до девятки, обозначенные цифрами от 1 до 9, причем выигрывает набравший первым три карты с суммой числовых значений 15. Связь этой игры с «крестиками — ноликами» становится очевидной, если клетки игровой таблицы пронумеровать, как показано на рис. 17, когда образуется так называемый магический квадрат с одинаковыми суммами номеров клеток, расположенных по горизонтали, вертикали или диагонали. Если теперь карты, разложенные в клетках такого магического квадрата, перенумеровать в соответствии с рис. 14, в, то в такой игре с картами может успешно участвовать и микрокалькулятор с программой 14, но для этого случая можно составить и более простую программу, воспользовавшись свойствами магического квадрата.

Условия рассмотренных игр близки к условиям игры Тригекс [3], два участника которой по очереди занимают вершины фигуры, показанной на рис. 18, причем выигрывает занявший первым три вершины, расположенные на одной прямой. Однако в такой игре число вариантов значительно больше, чем в «крестиках — ноликах», и сокращение числа перебираемых вариантов еще более желательно.

Для этого достаточно установить, что вершины 1, 3 и 4 перекрывают все прямые, и, следовательно, занятие этих вершин лишает противника возможности выиграть. Так как остальные вершины симметричны относительно этих вершин, то для первого хода достаточно выбрать одну из них (например, вершину 1), а для последующих ходов — такие вершины, занятие которых создаст угрозу проигрыша противнику и заставит его предотвращать эту угрозу, а не добиваться более выигрыш-

ного положения. При таком выборе ходов число рассматриваемых вариантов оказывается не очень большим и граф игры может быть непосредственно использован для программирования микро-ЭВМ. Примером может служить программа 15, построенная по такому графу (рис. 19).

Игра по программе 15 неизбежно приводит к победе микрокалькулятора, например $77 = РД$ В/О С/П $PX = 1; 8 = PX$ С/П $PX = 2; 5 = PX$ С/П $PX = 7; 9 = PX$ С/П $PX = 77; PY = 3$ В/О С/П $PX = 1; 4 = PX$ С/П $PX = 9; 8 = PX$ С/П $PX = 77, PY = 7$ и т. д.

Составляя алгоритм некоторых игр, можно избежать необходимости перебора всех вариантов, используя симметрию ходов. Примером является игра на шахматной доске с исходным положением пешек, показанным на рис. 20, а. Играющие по очереди передвигают свои пешки по вертикалям доски, не «перепрыгивая» через чужие пешки, с целью «запереть» все пешки противника (рис. 20, б). При нечетном числе игровых вертикалей начинающие игру белые обеспечивают себе победу, «заперев» первым

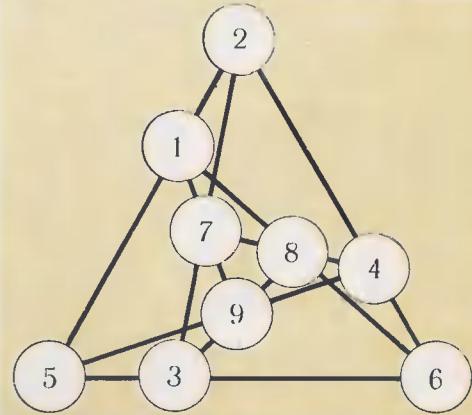


Рис. 18.

ходом одну из пешек противника и в дальнейшем на каждый ход противника по одной из вертикалей отвечая симметричным ходом по другой вертикали [13]. Эту игру уместно назвать «мушкетерами», движения которых при фехтовании напоминают перемещения пешек в игре.

Рассмотренный способ оптимальной игры белых в «мушкетеры» легко программировать при кодировании цифрами *xy* полей, занимаемых пешками при очередном ходе. Победа программируемого микрокалькулятора в этой игре обеспечивается программой 16.

Пример партии в «мушкетеры» с микрокалькулятором, «обученным» по программе 16: БП 4 0 С/П $PX = 37$; 25 = PX С/П $PX = 14$; 15 = PX С/П $PX = 24$; 17 = PX С/П $PX = 16$; 26 = PX С/П $PX = 25$; 18 = PX С/П $PX = 17$; 27 = PX С/П $PX = 26$; 28 = PX С/П $PX = 27$.

Правила многих игр с полной информацией однозначно определяют концевую позицию, соответствующую выигрышу одного участника, первым достигшего этой позиции, и проигрышу другого участника. Можно найти оптимальную стратегию (наилучший метод) такой игры, не при-

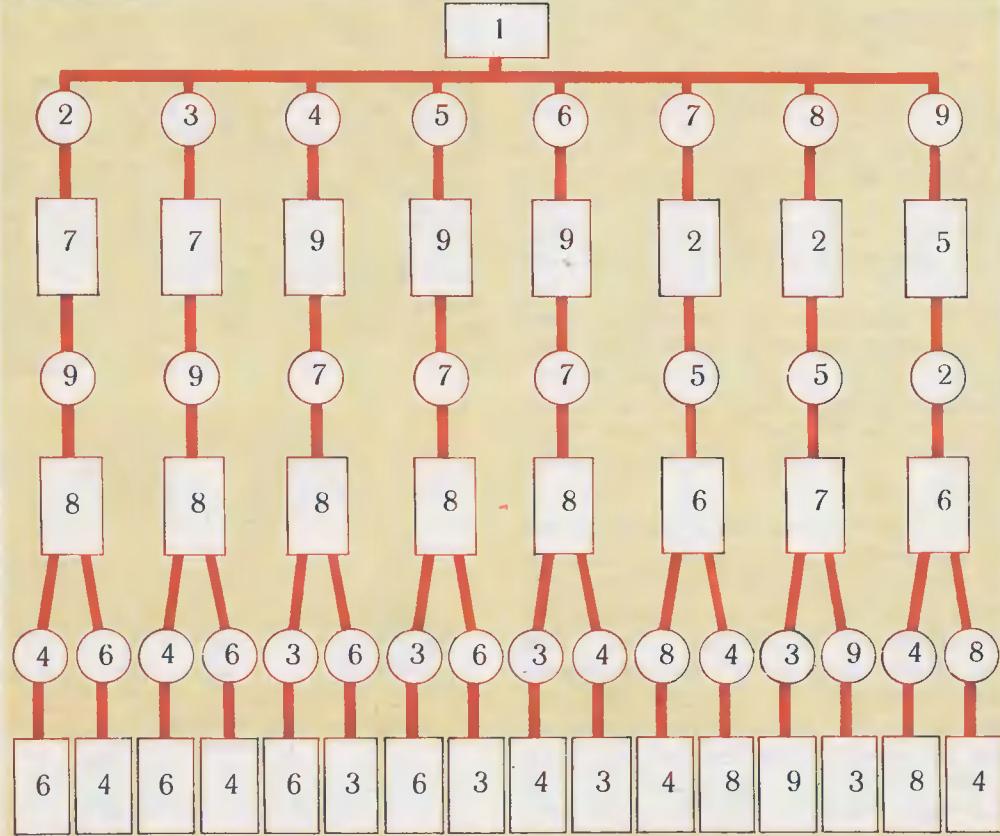


Рис. 19.

Программа 15. Игра Тригекс

Сх 0Г	1 01	С/П 50	П7 47	4 04	— 11	$x \geq 0$ 59	36 36	3 03	— 11
$x \geq 0$ 59	53 53	2 02	— 11	$x = 0$ 5E	74 74	5 05	С/П 50	П8 48	2 02
— 11	$x \neq 0$ 57	26 26	Вх 0	ИПД 6Г	С/П 50	6 06	С/П 50	П9 49	8 08
— 11	$x = 0$ 5E	23 23	4 04	ИПД 6Г	С/П 50	7 07	С/П 50	П8 48	9 09
— 11	$x = 0$ 5E	23 23	8 08	С/П 50	П9 49	4 04	— 11	$x = 0$ 5E	23 23
6 06	ИПД 6Г	С/П 50	9 09	С/П 50	П8 48	7 07	— 11	$x = 0$ 5E	23 23
8 08	С/П 50	П9 49	3 03	— 11	$x = 0$ 5E	23 23	ИП7 67	6 06	— 11
$x = 0$ 5E	23 23	БП 51	33 33	2 02	С/П 50	П8 48	5 05	— 11	$x = 0$ 5E
23 23	ИП7 67	7 07	— 11	$x = 0$ 5E	88 88	БП 51	26 26	7 07	С/П 50
П9 49	3 03	— 11	$x = 0$ 5E	23 23	9 09	ИПД 6Г	С/П 50		

Инструкция. ($77 = P\Delta$) В/О С/П $PX = 1$; $n_1 = PX$ С/П $PX = m_2$; $n_2 = PX$ С/П $PX = m_3$... С/П $PX = 77$ (выигрыш микрокаль-

кулятора); $PY = m_k$ (номер, вершины, занятой последним ходом микрокалькулятора).

бегая к полному перебору ее вариантов. Для этого следует решить задачу «с конца», определив для концевой позиции предыдущую и последующие позиции, называемые особыми, от каждой из которых можно достичь следующей особой позиции (и, в конечном итоге, концевой) только за два хода. Участник, начинающий игру из неособой позиции, выигрывает, достигая первым и последующими ходами особых позиций, тогда как второй участник, начинающий свой первый ход из особой позиции, может выиграть только при ошибке противника.

Одна из простейших игр с особыми позициями называется игрой Баше (по имени автора, описавшего ее еще в начале XVII столетия). Основной

вариант этой игры описывается следующими правилами: из множества S предметов (например, спичек или камешков) игроки по очереди забирают от одного до k предметов, причем выигрывает взявший последний предмет.

Особые позиции в этой игре соответствуют числу предметов, кратному $k + 1$. Следовательно, согласно упомянутой в предыдущей главе теории сравнений, для определения, особая ли позиция с S предметами, необходимо найти остаток от деления S на $k + 1$:

$$d = S - (k + 1) E(S/(k + 1)),$$

где E — символ целой части содержащего скобок.

Если $d \neq 0$, то позиция неособая, и участник, начинающий игру с этой

позиции, должен взять d предметов для перехода к особой позиции, а при последующих ходах должен брать $m = k + 1 - n$ предметов, где n — число предметов, взятых противником предыдущим ходом. Если же $d = 0$, то позиция особая и участнику, начинаяющему ход с этой позиции, целесообразно взять только один предмет, чтобы увеличить вероятность ошибоч-

ного хода противника, не занимающего особой позиции этим ходом.

Эта оптимальная стратегия для обоих участников игры Баше реализована в алгоритме, схема которого показана на рис. 21. В алгоритме предусмотрена установка флага ($f = 0$) при ходе из особой позиции, который снимается ($f = d$) при занятии особой позиции для последующих переходов

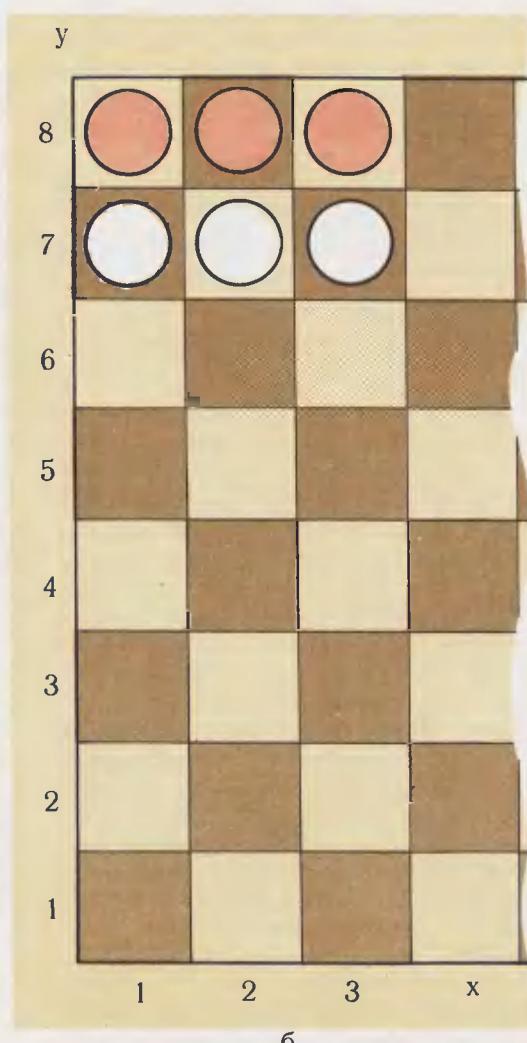
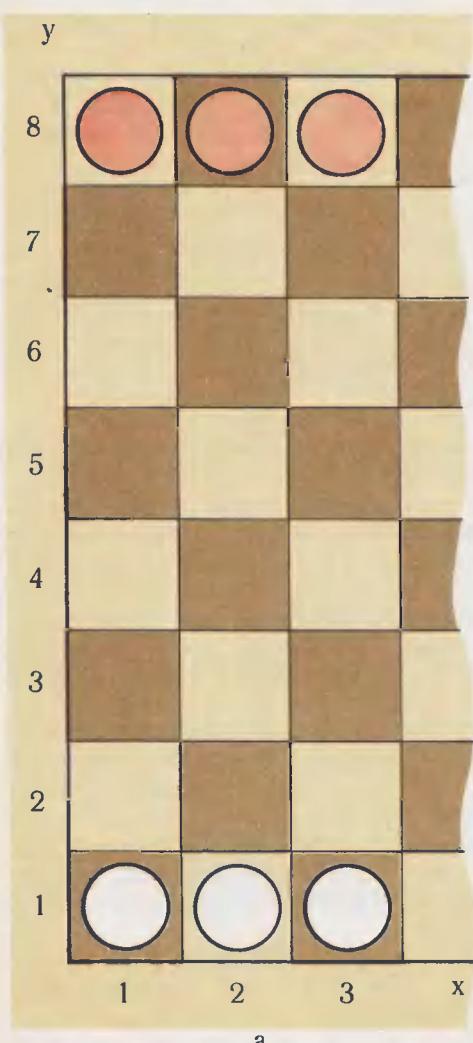


Рис. 20.

Программа 16. Игра белыми в «мушкетеры»

П9 49	2 02	0 00	— 11	$x < 0$ 5C	24 24	ИП1 61	ИП9 69	П1 41	— 11
$x \geq 0$ 59	18 18	ИП4 64	— 10	П4 44	С/П 50	БП 51	00 00	ИП3 63	ХУ 14
— 11	П3 43	БП 51	15 15	ИП2 62	ИП9 69	П2 42	— 11	$x \geq 0$ 59	35 35
ИП3 63	— 10	П3 43	БП 51	15 15	ИП4 64	ХУ 14	— 11	БП 51	14 14
Сх 0Г	1 01	8 08	П1 41	2 02	8 08	П2 42	1 01	1 01	П3 43
2 02	1 01	П4 44	3 03	7 07	БП 51	15 15			

Инструкция. Нажать клавиши БП 4 0 С/П, что приведет к высвечиванию первого хода белых на клетку $xy = 37$; после каждого хода микрокалькулятора вводить код xy клет-

только к особой позиции. На входном языке программируемых микрокалькуляторов этот алгоритм представлен программой 17.

Для проверки программы 17 следует выполнить: $3 = P8$; $13 = P9$; $0 = PX$ БП 5 0 С/П $PX = 1$ ($PY = 12$); $1 = PX$ С/П $PX = 3$ ($PY = 8$); $3 = PX$ С/П $PX = 1$ ($PY = 4$); $1 = PX$ С/П $PX = 0$ ($PY = 3$) (выиграл микро-

ки, занимаемой ходом черных, в регистр X и нажимать клавиши В/О и С/П или С/П; окончание игры определяется положением фигур на доске.

калькулятор, «взявший» последним ходом 3 предмета); $1 = PX$ БП 5 0 С/П $PX = 1$ ($PY = 11$); $3 = PX$ С/П $PX = 1$ ($PY = 7$); $3 = PX$ С/П $PX = 1$ ($PY = 3$); $3 = PX$ С/П $PX = PY = 0$ (микрокалькулятор проиграл).

Известны различные варианты игры Баше. Участники одного из них по очереди добавляют к множеству от одного до k предметов, причем

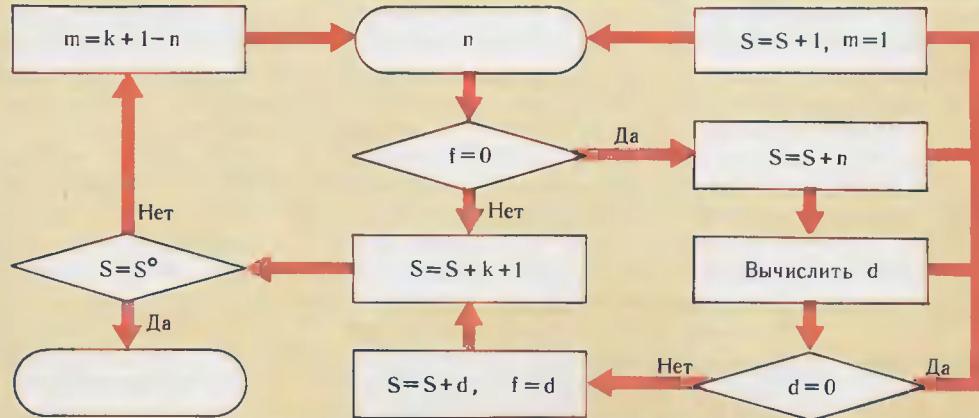


Рис. 21.

Программа 17. Игра Баше

П4 44	ИП7 67	$x = 0$ 5E	42 42	ИП6 66	ИП4 64	— 11	$x \neq 0$ 57	37 37	П6 46
ИП5 65	\div 13	1 01	+	10	П1 41	КИП1 Г1	ИП6 66	ИП1 61	ИП5 65
— 11	$x = 0$ 5E	30 30	ИП6 66	1 01	— 11	П6 46	1 01	БП 51	39 39
П7 47	ИП6 66	ИП7 67	— 11	П6 46	$x = 0$ 5E	38 38	С/П 50	ИП7 67	С/П 50
БП 51	00 00	ИП5 65	ИП4 64	— 11	П7 47	ИП6 66	ИП5 65	БП 51	33 33
П4 44	Сх 0Г	П7 47	ИП9 69	П6 46	ИП8 68	1 01	+	П5 45	ИП4 64
БП 51	00 00								

Инструкция. ($k = P8$, $S = P9$) ввести $0 = PX$ при первом ходе микрокалькулятора или число предметов, забираемых противником на первом ходу; $n = PX$ и нажать клавишу БП 5 0 С/П, что приведет к высвечиванию числа m предметов, «взятых» микрокалькулятором его первым ходом (в регистре Y хранится число оставшихся предметов); в дальнейшем

выигрывает первым дополнивший множество до заданного числа S предметов. Для участия микрокалькулятора в этом варианте игры также пригодна программа 17, но в регистре Y в этом случае хранится число предметов, которыми требуется дополнить множество до заданного числа S предметов.

В других вариантах игры Баше проигрывает взявший или положивший последний предмет. Для участия микрокалькулятора в таких играх также можно использовать программу 17, но перед началом игры в регистр 0 следует ввести число $S - 1$. Тогда высвечивание нуля будет означать

вводить в регистр X число предметов, взятых на очередном ходу противника, и нажимать только клавишу С/П, что будет приводить к высвечиванию числа предметов, «взятых» в ответ микрокалькулятором; игра заканчивается при высвечивании нуля выигрышем микрокалькулятора ($PY = m \neq 0$) или его противника ($PY = 0$).

чать выигрыш микрокалькулятора ($PY = 1$) или его противника ($PY = 0$), оставляющих сопернику один предмет, который должен быть взят.

Для игры Баше иногда используют игровую доску с последовательностью обозначенных порядковыми номерами клеток с максимальным номером S клетки, в которую перед началом игры помещают фишку (рис. 22). Игроκи по очереди передвигают фишку на несколько (но не более k) клеток с выигрышем игрока, первым достигшим последней клетки с номером 0. Эти условия, соответствующие основному варианту игры Баше, изменяют в соответствии с правилами других вариантов. В таких играх также может участвовать микрокалькулятор, «обученный» по программе 17.

Более сложная игра с особыми позициями для двух множеств предметов известна на Востоке под названием

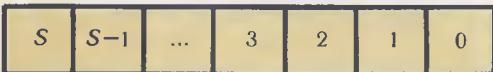


Рис. 22.

Цзяньшидзы, или «выбиранье камней» [13]. Участники этой игры по очереди берут произвольное (но не менее одного) число предметов из одного множества или одинаковое из обоих множеств, причем выигрывает взявший последний предмет. В этой игре, как показывает математический анализ [5], некоторые из позиций (a, b) с числами a и b предметов ($a \leq b$) в множествах являются особыми позициями (c_k, d_k) $_k = (0, 0)_0, (1, 2)_1, (3, 5)_2, (4, 7)_3, (6, 10)_4, (8, 13)_5, \dots$ с разностью $k = d_k - c_k = 0, 1, 2, 3, 4, 5, \dots$, причем c_k — наименьшее число, не содержащееся в предыдущих особых позициях.

Участник, начинаящий игру с неособой позиции, выигрывает, занимая на каждом ходу особую позицию. Если он начинает игру с особой позиции, то ему остается занимать позиции с $a \neq b$ в надежде на ошибку противника.

Для определения особенностей данной позиции (a, b) следует найти целое число k в интервалах $(a(\tau - 1); (a + 1)(\tau - 1))$ или $(a(2 - \tau); (a + 1) \times (2 - \tau))$, где $\tau = (1 + \sqrt{5})/2$. Если k содержится в первом интервале, то $a = c_k$, если во втором, то $a = d_k$. Если соответственно $a + k = b$ или $a = b - k$, то позиция (a, b) особая. В противном случае для перехода к особой позиции руководствуются следующими правилами:

1. Если $a = b$, то следует взять по a предметов из каждого множества и перейти к выигрышной позиции $(0, 0)$.

2. Если $a = c_k$, $b > a + k$, то следует взять из второго множества число предметов $b - a - k$, получив особую позицию $(a, a + k)$.

3. Если $a = c_k$, $b < a + k$, то следует взять от обоих множеств по $a - q$ предметов, где $q = E((b - a)\tau)$; символом E обозначена целая часть содержащего квадратных скобок.

4. Если $a = d_k$, то следует взять из второго множества $b - c_k$ предметов, где $c_k = E[k\tau]$.

В соответствии с этими правилами, составим следующий алгоритм для выбора очередного хода по данной позиции (a, b) при $a \leq b$:

1. Если $E[(a + 1)(\tau - 1)] - E[a(\tau - 1)] \geq 1$, то перейти к шагу 2, иначе — к шагу 8.
2. Принять $k = E[(a + 1)(\tau - 1)]$.
3. Если $a + k = b$, то перейти к шагу 4, иначе — к шагу 5.
4. Принять $a = a$, $b = b - 1$.
5. Если $a + k < b$, то перейти к шагу 6, иначе — к шагу 7.
6. Принять $a = a$, $b = b - E[k(1 + \tau)]$.
7. Принять $q = E[(b - a)\tau]$, $a = q$, $b = b - a + q$.
8. Если $a + k > b$, то перейти к шагу 4, иначе — к шагу 9.
9. Принять $k = E[(a + 1)(2 - \tau)]$, $a = E[k\tau]$, $b = a$.

При реализации этого алгоритма на входном языке программируемых микрокалькуляторов следует учитывать, что величины $a(\tau - 1)$ и $(a + 1) \times (\tau - 1)$ могут быть меньшими единицы и для выделения их целой части следует использовать фрагмент вида $1 + \text{ПН КИПН}$ с адресным регистром $N < 4$. В этом случае при выполнении оператора КИПН целое содержимое регистра N уменьшается на единицу, что компенсирует предварительное увеличение на единицу исходного числа. Это учтено в программе 18 с дополнительным фрагментом для автоматического вычисления $\tau = (\sqrt{5} + 1)/2$.

Правильность работы по программе 18 можно проверить, повторив следующую партию: $300 = PY; 400 = = PX \text{ B/O C/P } PX = 261; PY = 161; 100 = PY; 200 = PX \text{ B/O C/P } PX = = 100; PY = 162; 10 = PY; 162 = PX \text{ B/O C/P } PX = 10; PY = 6; 1 = PY; 6 = PX \text{ B/O C/P } PX = 2; PY = 1;$

Программа 18. Игра «выбиранье камней»

П8 48	ХУ 14	П7 47	1 01	+	10	5 05	$\sqrt{}$	22	1 01	+	10	2 02
\div 13	П9 49	1 01	— 11	\times	12	1 01	+	10	П1 41	КИП1 Г1	ИП7 67	
ИП9 69	1 01	— 11	\times	12	1 01	+	10	П2 42	КИП2 Г2	ИП1 61	ИП2 62	
— 11	$x \neq 0$ 57	55 55	ПП 53	76 76	$x < 0$ 5C	42 42		ИП7 67	ИП7 67	ИП1 61		
+	10	С/П 50	ИП8 68	ИП7 67	— 11	П6 46	ИП9 69	\times	ПА 4—	КИПА Г—		
ИПА 6—	ИП6 66	ИПА 6—	+	10	С/П 50	ИП7 67	1 01	+	10	2 02	ИП9 69	
— 11	\times	12	1 01	+	10	П1 41	КИП1 Г1	ПП 53	76 76	ИП1 61	ИП9 69	
\times	12	ПА 4—	КИПА Г—	ИПА 6—	ИП7 67	С/П 50	ИП7 67	ИП1 61	+	10	ИП8 68	
— 11	$x = 0$ 5E	88 88	ИП7 67	ИП8 68	1 01	— 11	С/П 50	В/О 52				

Инструкция. $a = PY$, $b = PX$ (должно быть $a \leq b$); В/О С/П $PX = b$; $PY = a$ (при выигрыше микрокалькулятора $PX = PY = 0$;

при проигрыше микрокалькулятора $PX = PY = 1$).

$$0 = PY; \quad 2 = PX \quad \text{В/О} \quad \text{С/П} \quad PX = PY = 0.$$

Для этой игры можно также использовать игровую доску, подобную показанной на рис. 22, с двумя фишками, расположенными перед началом игры в клетках доски с номерами a и b . Игроки по очереди передвигают обе фишки на одинаковое число клеток или одну фишку на произвольное (но не менее единицы) число клеток. Выигрывает игрок (которым может быть и микрокалькулятор, «обученный» по программе 18), после хода которого обе фишки оказываются в клетке с номером 0.

Возможны и другие варианты игры «выбиранье камней», например, с проигрышем игрока, взявшего последний предмет, или с поочередным добавле-

нием предметов в двух множествах до заданного количества. Примером может служить игра «одинокий ферзь» [13]. Для участия в этой игре микрокалькулятора с программой 18 следует изменить нумерацию клеток доски (учитывая, что изменяющимся числам предметов соответствуют номера горизонталей и вертикалей шахматной доски) или составить специальную программу.

В исходном положении этой игры противник располагает ферзя в одной из девяти клеток нижнего левого угла доски ($x \leq 3$, $y \leq 3$), и участники по очереди передвигают ферзя вверх, вправо или по диагонали вверх вправо до достижения выигрышной клетки с кодом $xy = 88$ (рис. 23). В этой игре клетки шахматной доски с ко-

дами $xy = 88, 76, 67, 53$ и 35 соответствуют особым позициям $(0, 0)$, $(1, 2)$ и $(3, 5)$ игры «выбиение камней», но программу для участия в ней микро-ЭВМ проще составить методом перебора вариантов ходов. Успешное участие в такой игре программируемого микрокалькулятора обеспечивается программой 19.

Время выполнения этой программы зависит от положения ферзя, так как

перебирается различное число клеток, например, в партии: $13 = PX$ БП 5 7 С/П $PX = 35$ (время счета около 30 с); $65 = PX$ С/П $PX = 76$ (время счета около 15 с); $87 = PX$ С/П $PX = 88$ (время счета около 75 с).

Если изменить правила игры «выбиение камней», разрешив игрокам уменьшать или увеличивать оба или одно множество предметов только на единицу, то такой вариант соответ-

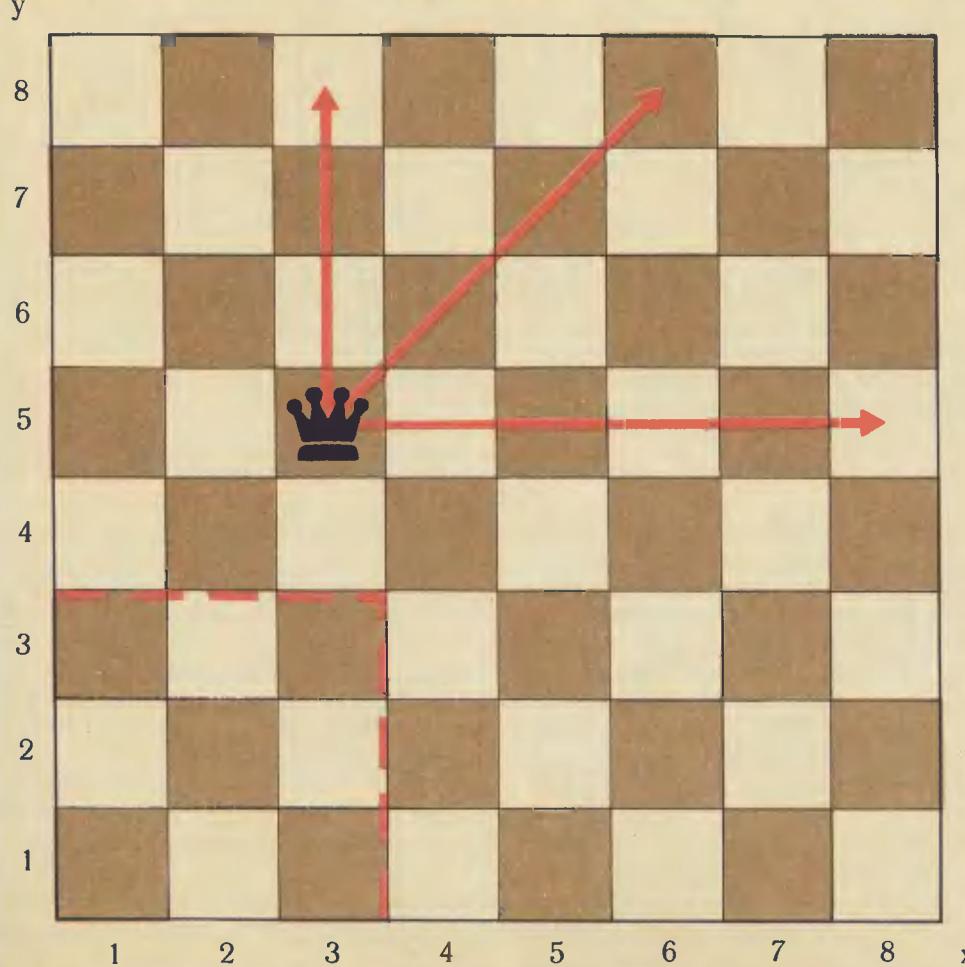


Рис. 23.

Программа 19. Игра «одинокий ферзь»

П7 47	П9 49	3 03	П6 46	6 06	П0 40	ИП9 69	1 01	1 01	ПП 53
40 40	x = 0 5E	04 04	ИП7 67	П9 49	4 04	П6 46	4 04	П0 40	ИП9 69
1 01	ПП 53	40 40	x = 0 5E	17 17	ИП7 67	П9 49	4 04	П6 46	4 04
П0 40	ИП9 69	1 01	0 00	ПП 53	40 40	ИП9 69	С/П 50	БП 51	00 00
+	П9 49	КИП0 Г0	ИП9 69	— 11	x ≠ 0 57	36 36	ИП0 60	1 01	— 11
x = 0 5E	42 42	ИП6 66	1 01	— 11	П6 46	В/О 52	П7 47	П9 49	7 07
П6 46	8 08	8 08	П1 41	7 07	6 06	И2 42	6 06	7 07	П3 43
5 05	3 03	П4 44	3 03	5 05	П5 45	БП 51	04 04		

Инструкция. Ввести в регистр X код xy клетки, занимаемой ферзем в исходном положении ($x \leq 3, y \leq 3$) и нажать клавиши БП 5 7 С/П для высвечивания кода xy клетки, на которую микрокалькулятор «передвигает» ферзя; вводить код xy клетки, на которую пере-

двигается ферзь, и нажимать клавиши В/О или С/П или С/П для регистрации ответного хода микрокалькулятора; игра заканчивается при занятии микрокалькулятором клетки с кодом $xy = 88$.

Программа 20. Игра «одинокая пешка»

↑ 0Е	1 01	0 00	÷ 13	П7 47	КИП7 Г7	→ 25	ИП7 67	— 11	1 01
0 00	×	12	П8 48	ПП 53	61 61	ПВ 4L	ИП7 67	ПП 53	61 61
ИПВ 6L	+	10	x ≠ 0 57	41 41	ИП7 67	ИПА 6—	+	10	П7 47
+	П8 48	ИП8 68	ИП7 67	1 01	0 00	×	12	+	С/П 50
00 00	ИП7 67	ИП8 68	— 11	x ≠ 0 57	55 55	ИП8 68	8 08	— 11	x ≠ 0 57
55 55	ИП8 68	1 01	+	10	П8 48	ИП7 67	1 01	+	10
32 32	2 02	— 11	x ≠ 0 57	69 69	x < 0 5C	61 61	2 02	+	10
									В/О 52

Инструкция. Ввести в регистр X код $xy = 11$ при первом ходе микрокалькулятора или код xy клетки, занимаемой первым ходом противника, и нажать клавиши В/О и С/П; после высвечивания кода клетки, занимаемой

микрокалькулятором, вводить код клетки, занимаемой противником, и нажимать клавиши В/О и С/П или С/П; выигрывает участник игры, первым занявшим поле шахматной доски с кодом $xy = 88$ (h8).

Программа 21. Игра Ним с тремя множествами предметов

ИП8 68	ИП9 69	— 11	$x = 0$ 5E	07 07	П7 47	С/П 50	7 07	П1 41	1 01
0 00	ПА 4—	2 02	ПВ 4L	ИП7 67	ПП 53	67 67	ПС 4C	ИП8 68	ПП 53
67 67	ИПС 6C	— 10	1 01	ВП 0C	7 07	÷ 13	1 01	0 00	× 12
П5 45	1 01	— 10	П0 40	КИПО Г0	ИП0 60	2 02	— 11	$x \neq 0$ 57	41 41
ИП0 60	ИПД 6Г	1 01	0 00	× 12	+	10	ПД 6Г	ИП5 65	ИП0 60
L1 5L	27 27	ИПА 6—	ПВ 4L	2 02	ПА 4—	ИПД 6Г	ПП 53	67 67	ИП9 69
— 11	$x = 0$ 5E	64 64	КИПЗ Г3	ИПЗ 63	П9 49	С/П 50	ХУ 14	Сх 0Г	П3 43
1 01	Вх 0	↑ 0Е	ИПВ 6L	÷ 13	1 01	+	10	П0 40	КИП0 Г0
→ 25	ИП0 60	ИПВ 6L	× 12	— 11	×	12	ИП3 63	+	10
ИПА 6—	× 12	ИП0 60	$x = 0$ 5E	72 72	ПД 4Г	ИП3 63	В/О 52		

Инструкция. Перед каждым пуском программы $a = P7$, $b = P8$, $c = P9$ ($a \leq b \leq c$) и нажать клавиши В/О и С/П; после выполнения программы содержимое одного из регистров 7, 8 или 9 изменяется (изменяемое со-

держимое регистра хранится также в регистре X); при выигрыше микрокалькулятора $P7 = P8 = P9 = 0$, при проигрыше в одном из регистров 7, 8 или 9 хранится единица, в остальных — нули.

ствует игре «одинокая пешка» [7] на шахматной доске. Участники игры по очереди передвигают пешку (находящуюся в исходном положении на поле a1) на одно поле вверх, вправо или вверх вправо по диагонали, причем выигрывает первым достигший поля h8. При кодировании полей доски цифровым кодом xy особые позиции в этой игре соответствуют полям с четными номерами x и y (отмеченные в кружочках крестиками на рис. 24).

Оптимальная стратегия первого участника этой игры заключается в занятии первым ходом поля с кодом $xy = 22$ с последующим переходом только на особые позиции. Второму

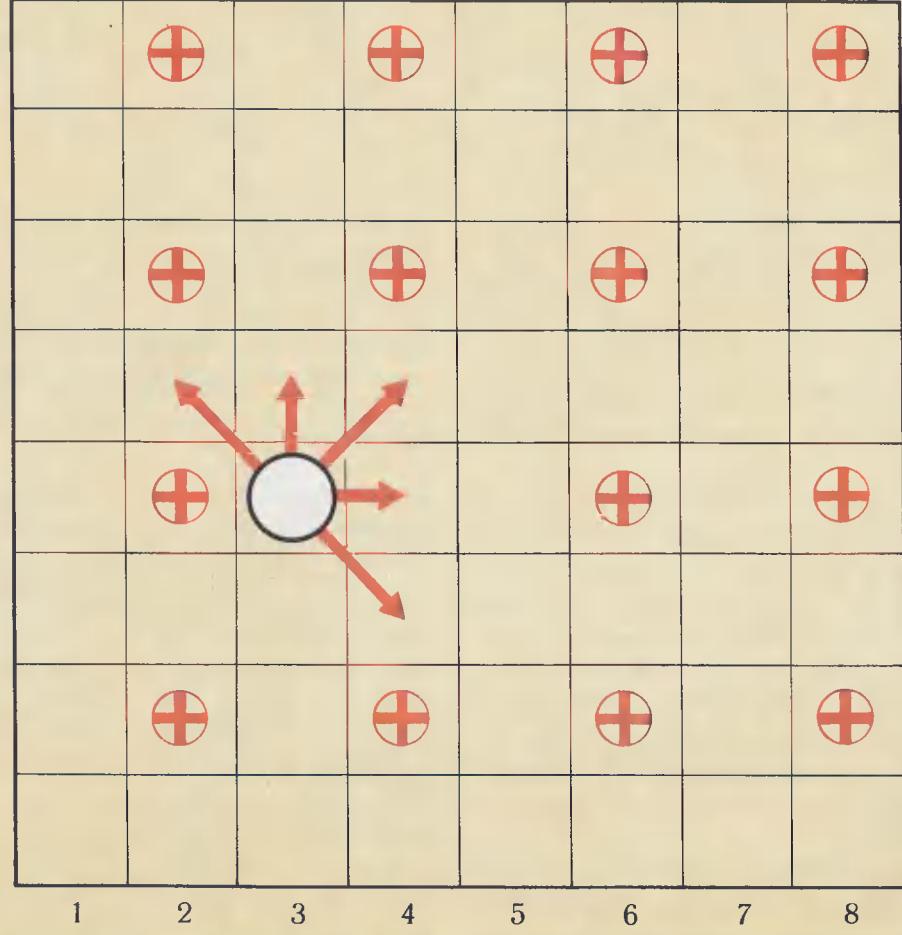
игроку следует сойти с главной диагонали и двигаться, когда это возможно, параллельно ей в надежде на ошибку противника. Эта стратегия реализована в алгоритме, схема которого показана на рис. 25, содержащем операторы для установки и проверки флага при занятии особой позиции. На входном языке микрокалькуляторов алгоритм реализуется программой 20.

Пример партии (время «размышления» микрокалькулятора над каждым ходом около 25 с): $22 = PX$ В/О С/П $PX = 32$; $43 = PX$ С/П $PX = 44$; $54 = PX$ С/П $PX = 64$; $75 = PX$ С/П $PX = 86$; $87 = PX$ С/П $PX = 88$ (выиграл микрокалькулятор).

В Европе давно известна более сложная игра Ним с тремя или более множествами предметов. Два участника основного варианта этой игры по очереди берут из любого одного множества произвольное (но не менее одного) число предметов, причем выигрывает взявший последний предмет. Особую популярность эта игра приобрела среди программистов. Согласно теории игры [5], при числе

множеств три или более следует сложить двоичные представления чисел предметов в множествах по правилам сложения десятичных чисел. Если все цифры суммы четные, то этому числу предметов в множествах соответствует особая позиция. В противном случае для перехода к особой позиции следует сформировать двоичное представление d_2 , заменив в полученной сумме четные цифры нулями,

у



х

Рис. 24.

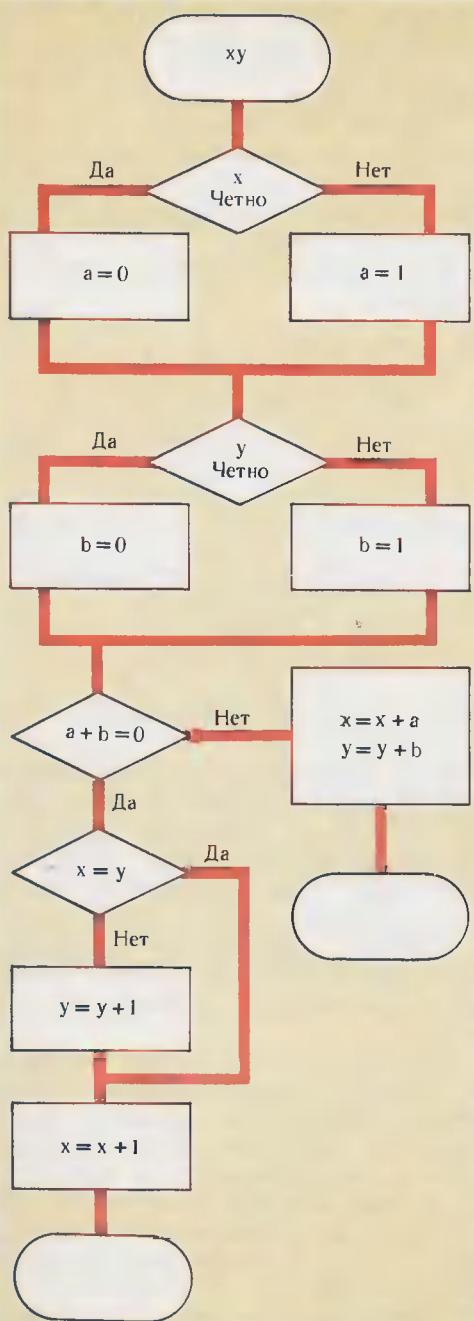


Рис. 25.



Рис. 26.

а нечетные — единицами, и найти его десятичное представление d_{10} . После этого достаточно уменьшить одно из чисел предметов в множествах на величину d_{10} . Следовательно, оптимальный алгоритм игры Ним при выборе очередного хода в позиции (a, b, \dots, z) можно представить следующим описанием:

1. Найти двоичные представления чисел предметов в множествах.

2. Сложить эти представления по правилам сложения десятичных представлений чисел.

3. Сформировать число d_2 , заменив в сумме четные цифры нулями и нечетные единицами.

4. Если $d_2 = 0$, то перейти к шагу 5, иначе — к шагу 6.

5. Уменьшить на единицу число предметов в одном из множеств.

6. Найти десятичное представление d_{10} двоичного представления d_2 .

7. Уменьшить на величину d_{10} число предметов в одном из множеств для перехода к особой позиции.

Наиболее громоздкие операции в этом алгоритме связаны с преобразованием двоичных представлений чисел в десятичные и десятичных в двоичные. Такие операции можно реализовать с помощью алгоритма, представленного программой 1. Для наиболее распространенного варианта игры Ним с тремя множествами предметов выбор оптимального хода в позиции (a, b, c) , где $a \leq b \leq c$, обеспечивается алгоритмом, схема которого показана на рис. 26. На входном языке микрокалькуляторов этот алгоритм представлен программой 21, при использовании которой максимальное исходное число предметов в множествах $2^8 - 1 = 255$ ограничено разрядностью мантиссы операндов в микрокалькуляторах.

Для игры Ним также можно воспользоваться игровой доской, подобной показанной на рис. 22, с ис-

ходным положением трех фишек в клетках с номерами a, b и c и поочередным передвижением каждым игроком одной из фишек.

В связи с громоздкостью преобразований, выполняемых по этой программе, микрокалькулятор затрачивает на «обдумывание» каждого хода несколько минут. Пример партии с микрокалькулятором по этой программе:

$20 = P7, 40 = P8, 60 = P9$ В/О С/П
 $PX = P9 = 59, P7 = 20, P8 = 40;$
 $30 = P8, 40 = P9$ В/О С/П
 $PX = P9 = 10, P7 = 20, P8 = 30;$
 $20 = P8 = P9$ В/О С/П $PX = P7 = 0,$
 $P8 = 20, P9 = 20;$
 $10 = P8$ В/О С/П $PX = P9 = 10,$
 $P7 = 0, P8 = 10;$
 $5 = P8$ В/О С/П $PX = P9 = 5,$
 $P7 = 0, P8 = 5;$
 $1 = P8$ В/О С/П $PX = P9 = 1,$
 $P7 = 0, P8 = 1;$
 $0 = P8$ В/О С/П
 $PX = P7 = P8 = P9 = 0.$

При использовании микро-ЭВМ с достаточно большой емкостью памяти по алгоритмам, подобным рассмотренным, можно составить программы для игры Ним с большим числом множеств предметов, а также для игр с близкими к игре Ним правилами, например игры «кегли» [3] с разбиением множеств предметов в процессе игры.

Правила некоторых игр с особыми позициями предусматривают выполнение победителем дополнительных условий. Примером может служить игра с одним множеством предметов, участники которой по очереди берут от одного до k предметов, но выигрывает тот участник, который набрал четное или нечетное (по договоренности) их число. Выбор оптимального хода в таких играх зависит от числа k и условий выигрыша, поэтому во мно-

Программа 22. Игра со спичками

П8 48	Сх ОГ	П2 42	П3 43	/—/ 0L	П9 49	4 04	+	10	$x \geq 0$ 59	61	61
ИП9 69	ИП8 68	+	10	$x \geq 0$ 59	61 61	П8 48	$x \neq 0$ 57	54 54	ИП3 63	ИП9 69	
— 11	П3 43	2 02	— 11	$x < 0$ 5C	22 22	1 01	+	10	$x \neq 0$ 57	26	26
ИП8 68	+	10	П5 45	6 06	— 11	$x < 0$ 5C	32 32	ИП5 65	5 05	—	11
$x \geq 0$ 59	44 44	4 04	П5 45	ИП5 65	ИП2 62	+	10	П2 42	ИП5 65	ИП8 68	
— 11	$x \geq 0$ 59	58 58	/—/ 0L	↑ 0E	ИП2 62	+	10	C/П 50	/—/ 0L	П8 48	
ИП5 65	C/П 50	БП 51	04 04								

Инструкция. Ввести нечетное число S в регистр X и нажать клавиши В/О и С/П (если высвечивается 0, то ввести в регистр X другое нечетное число S); после каждого выполнения программы высвечивается число спичек, «взятых» микрокалькулятором, а оставшееся число спичек хранится в ре-

гистре Y ; противник вводит в регистр X число взятых им спичек и нажимает клавишу С/П; при выигрыше микрокалькулятора высвечивается четное число «набранных» им спичек, а в регистре Y хранится число спичек, «взятых» последним ходом микрокалькулятора.

гих случаях оптимальную стратегию находят методом перебора возможных ходов. Примером может служить программа 22, составленная методом перебора для игры с множеством предметов (например, спичек) при $k = 4$ и выигрыше набравшего четное число предметов.

Микрокалькулятор, «обученный» по программе 22, «ведет себя» доста-

точно придирчиво — не начинает игру с невыгодного для него особого значения $S = 6i + 1$, не «берет» спичек, если не берет их противник, а при нарушении правил игры противником, взявшим большее положительного числа спичек, высвечивает отрицательное число, и игру можно продолжить лишь в соответствии с правилами.

Глава 5

ИГРЫ С НЕПОЛНОЙ ИНФОРМАЦИЕЙ

Во многих практических задачах решение приходится принимать до того, как станут точно известными действия противника, которыми могут быть и силы природы. Такие конфликтные ситуации моделируют *играми с неполной информацией*, участники которых при выборе оптимальной стратегии должны оценить вероятность и последствия каждого из возможных ходов противника. Так как для такой оценки часто используют таблицы (матрицы) с возможными результатами сочетаний своих стратегий и стратегий противника, то подобные игры называют также *матричными*.

Конфликтные ситуации, моделируемые играми с неполной информацией, часто встречаются в реальной жизни. Примером может служить следующий случай с тремя траулерами рыболовецкого совхоза. Их капитаны ожидали появления крупного косяка рыбы с равной вероятностью в северном (стратегия рыбы Р1) или южном (стратегия рыбы Р2) районах лова. По географическим и погодным условиям в южном районе лов рыбы мог продолжаться три дня, в северном — только один день.

Капитаны располагали четырьмя возможными стратегиями сосредоточения траулеров: два на севере и один на юге (стратегия К1), один на севере и два на юге (стратегия К2), все на севере (стратегия К3) и все на юге (стратегия К4). Стратегии К3 и К4 отброшены как чрезмерно рискованные, так как грозили возможностью

полной потери улова. Из двух оставшихся была выбрана стратегия К1, обеспечивающая наибольший улов при появлении косяка рыбы в любом из районов лова. Правильность выбранного капитанами решения подтверждается и теорией игр.

Теоретически любую игру с неполной информацией для двух участников можно описать на каждом ходу таблицей (матрицей) с n строками, соответствующими стратегиям x_i первого участника, и m столбцами, соответствующими стратегиям y_j второго участника. В каждой клетке такой таблицы на пересечении i -й строки и j -го столбца записывают выигрыш a_{ij} (равный проигрышу с отрицательным знаком) первого игрока при выборе им i -й стратегии и выборе противником j -й стратегии. Если выигрыш одного участника равен проигрышу другого, то их сопоставление называют игрой с нулевой суммой.

Наибольшие из наименьших элементов строк матрицы игры называют нижней чистой ценой игры, или *максимином* α , а наименьшие из наибольших элементов столбцов матрицы — верхней чистой ценой игры, или *минимаксом* β . Так как выигрыш первого участника игры не превышает проигрыша второго, то в общем случае $\alpha \leq \beta$. Если максимин и минимакс расположены в одной клетке матрицы, то ее содержимое называют *седловой точкой* игры, $\alpha = \beta$ — чистой ценой игры, соответствующей оптимальной стратегии обеих сторон. Игру

называют справедливой, если ее чистая цена равна нулю. Стратегии, обеспечивающие выигрыш при любых стратегиях противника, называют доминирующими.

В рассмотренной игре «капитаны» их выигрыш (и проигрыш рыбного косяка) можно оценить произведением числа дней лова на число траулеров в данном районе. Составляя матрицу игры с элементами, равными этим произведениям (табл. 3), легко определить седловую точку $\alpha = \beta = 2$, соответствующую оптимальной стратегии капитанов, являющейся доминирующей.

Во многих матричных играх седловая точка отсутствует, и доминирующую стратегию в таком случае найти невозможно. Оптимальная стратегия участников такой игры заключается в изменении на очередном ходу выбора чистых стратегий так, чтобы обеспечить появление седловой точки в матрице, отображающей многоходовую игру [13].

Оптимальную смесь солями p_{xi} и p_{yi} чистых стратегий x_i первого и y_j второго участников матричной игры можно определить при дополнении исходной матрицы для n и m чистых стратегий участников строками с элементами

$$a'_{ij} = p_{y1}(p_{x1}a_{11} + \dots + p_{xn}a_{n1}) + \dots + p_{ym}(p_{x1}a_{1m} + \dots + p_{xn}a_{nm})$$

и столбцами с элементами

$$a''_{ij} = p_{x1}(p_{y1}a_{11} + \dots + p_{ym}a_{1m}) + \dots + p_{xn}(p_{y1}a_{m1} + \dots + p_{ym}a_{mn}),$$

где p_{xi} и p_{yi} — предполагаемые оптимальными доли чистых стратегий, а a_{ij} — элементы исходной матрицы с чистыми стратегиями.

Если в этих дополнительных строках и столбцах содержится седловая точка, то соответствующие ей смешанные стратегии будут оптимальными. При оптимальной смеси стратегий первого участника средний выигрыш не зависит от выбора стратегии про-

3. Матрица игры «капитаны»

	P1	P2	α
K1	2	3	2
K2	1	6	1
β	2	6	

тивником. Поэтому для игр с матрицей 2×2 (с двумя чистыми стратегиями у каждого из двух участников) доля p_{x1} чистой стратегии x_1 первого участника в оптимальной смеси определяется из условия равенства среднего выигрыша при обеих стратегиях $a_{11}p_{x1} + a_{21}(1 - p_{x1}) = a_{12}p_{x1} + a_{22}(1 - p_{x1})$. Из этого равенства следует $p_{x1} = (a_{22} - a_{21})/(a_{11} + a_{22} - a_{21} - a_{12})$. Аналогично доля p_{y1} стратегии y_1 второго участника определяется как $p_{y1} = (a_{11} - a_{12})/(a_{11} + a_{22} - a_{21} - a_{12})$. Цена игры в этом случае соответствует оптимальной смеси стратегий и равна $\bar{A} = a_{11}p_{x1} + a_{12}(1 - p_{x1})$.

В играх с природой чистые стратегии в оптимальной пропорции можно выбирать на каждом ходу по определенному закону, но в игре с людьми должна быть обеспечена случайность выбора чистой стратегии с требуемой вероятностью, так как противник может воспользоваться замеченной закономерностью в смене стратегий для выигрыша.

Выбор решения человеком является волевым актом, и поэтому он не может выбирать стратегии по случайному закону с требуемой вероятностью. Для этой цели человек использует различные модели случайных событий с определенной вероятностью исходов — бросание монеты или кубика, вращение диска с цифрами или случайный выбор шаров, различных по цвету или нумерации.

Способы моделирования случайных событий с помощью микро-ЭВМ зависят от возможностей ее входного языка. При использовании программируемых микрокалькуляторов эту

ИП9 69	П0 40	КИПО ГО	ИП0 60	$x = 0$ 5E	02 02	БП 51	00 00
-----------	----------	------------	-----------	---------------	----------	----------	----------

Инструкция. ($k + 1 = P9$) В/О С/П, через 20—30 с (или более при $k > 10$) нажать клавишу С/П для аварийной остановки и вы-

свечивания номера исхода; при высвечивании нуля повторить выполнение инструкции с нажатия клавиш В/О и С/П.

задачу в простейших случаях можно решить с помощью программы без оператора остановки С/П, обеспечивающей хранение в регистре X номеров исходов случайного события в течение промежутков времени, пропорциональных заданной вероятности исходов. Тогда при аварийной остановке программы в случайный момент времени будет высвечиваться с заданной вероятностью номер одного из исходов моделируемого случайного события.

Подобный прием использован в программах 7 и 10, дополненных мини-программой вида Сх ПН КИПН БП А с адресом А перехода к оператору КИПН при номере адресного регистра $N = 4, 5$ или 6 . После пуска такой программы в регистре N формируются натуральные числа 1, 2, 3, ..., и аварийная остановка программы в случайный момент времени обеспечивает случайное содержимое регистра N . Однако такую программу можно использовать лишь как вспомогательную, так как формируемое натуральное число пропорционально времени выполнения программы. Поэтому для моделирования случайного события с k равновероятными исходами целесообразно использовать программу, подобную программе 23 с оператором косвенного вызова из памяти на адресном регистре с номером $N < 4$.

После пуска программы 23 в регистре O формируются целые числа $k, k - 1, \dots, 2, 1, 0, k, k - 1, \dots$, и аварийная остановка программы че-

рез достаточно длинный отрезок времени обеспечивает случайный выбор одного из исходов моделируемого случайного события. Так как число 0 хранится в регистре O несколько дольше других чисел (на время выполнения дополнительных четырех операторов), то вероятность высвечивания этого числа больше, чем других чисел, и его следует исключить при нумерации исходов случайного события.

При небольшом числе k исходов случайного события его моделирование упрощается. Так, случайное событие с двумя равновероятными исходами, обозначаемыми символами 0 и 1 (например, падение монеты вверхaversом или реверсом) моделируется мини-программами XY XY В/О или XY БП ОО с предварительным занесением цифр 0 и 1 в регистры X и Y . После каждого пуска такой программы нажатием клавиш В/О и С/П начинается обмен содержимым регистров X и Y (моделирующий, в частности, вращение бросаемой монеты) и при аварийной остановке в случайный момент времени высвечивается символ 0 или 1 с равной вероятностью $p(1) = 1 - p(0) = 0,5$.

Для моделирования случайного события с тремя или четырьмя равновероятными исходами можно использовать программу → → В/О или → БП 00 с предварительным занесением номеров исходов 0, 1, 2, 3 или 1, 2, 3, 4 (при высвечивании нуля испытание повторяют) в регистры операционного стека. Эти же мини-

программы пригодны для моделирования случайного события с вероятностью 0, 25; 0,5 или 0,75 одного из двух возможных исходов. Для этого достаточно предварительно занести в регистры операционного стека цифры 0 и 1 (или другие символы исходов) в требуемой пропорции.

Случайное событие с большим числом исходов можно также моделировать занесением в программную память в требуемой пропорции операторов набора символов исходов (программа 24) или вызова их из памяти.

Кстати, высвечивание нуля при выполнении программы 24 можно рассматривать как падение кубика на неровную поверхность, когда нельзя точно определить номер его верхней грани. Если же устраниТЬ в этой программе оператор Сх, то нуль высвечиваться не будет, но в этом случае цифра 5 будет храниться в регистре X несколько дольше других цифр, что нарушит их равновероятность.

Подобным образом можно моделировать и различные игры-лотереи по образцу программы 25. Эта програм-

Программа 24. Моделирование падения кубика

↑ 0E	6 06	↑ 0E	1 01	↑ 0E	4 04	↑ 0E	2 02	↑ 0E	3 03
↑ 0E	5 05	↑ 0E	Cx 0Г	BП 51	00 00				

Инструкция. Для моделирования исходов бросания кубика нажать клавиши В/О и С/П и через 15—20 с нажать клавишу С/П,

ма моделирует лотерею, в которой на каждые 18 билетов стоимостью по 1 коп. в среднем приходится выигрыш в 17 коп. Следовательно, на каждом билете микрокалькулятор «выигрывает», а его противник проигрывает $17/18$ коп., а вероятность выигрыша противником составляет $0,5 \cdot 17/18 = 0,472$. В действительности вероятность выигрыша еще меньше, так как при выполнении безусловного перехода регистр X очищен, и остановка программы в это время увеличивает выигрыш микрокалькулятора. В игре по этой программе можно убедиться, что при небольшом числе попыток можно не только «вернуть деньги», а и выиграть, но с увеличением числа попыток сумма проигрыша будет увеличиваться.

При выполнении рассмотренных программ моделирования случайных событий не следует пытаться угадать высвечиваемые цифры по индикатору или останавливать программу не в случайные моменты времени. Общий недостаток этих программ — невозможность их использования в качестве фрагментов программ гене-

тического редактора, что приведет к высвечиванию номера на верхней грани упавшего кубика; при высвечивании нуля повторить выполнение программы.

Программа 25. Модель лотереи с вероятностью выигрыша 0,472

1 01	Cx 0Г	1 01	Cx 0Г	3 03	Cx 0Г	1 01	Cx 0Г	1 01	Cx 0Г
3 03	Cx 0Г	1 01	Cx 0Г	5 05	Cx 0Г	1 01	Cx 0Г	BП 51	00 00

Инструкция. Покупка билета за 1 коп. моделируется нажатием клавиш В/О и С/П; после аварийной остановки программы наж-

тием клавиши С/П или другой клавиши высвечивается выигрыш 5, 3, 1 или 0 коп.

рирования случайных чисел с автоматической остановкой.

Во входных языках многих универсальных микро-ЭВМ имеются специальные команды (подобные команде *RND* языка БЕЙСИК или команде СЧ языка микрокалькуляторов «Электроника МК-61» и «Электроника МК-52») для формирования псевдослучайных (квазислучайных) чисел с равномерным распределением в интервале (0, 1). При отсутствии таких команд во входном языке их приходится реализовать с помощью более сложных фрагментов программы.

Выполнение машиной любой программы строго детерминировано, и при тех же исходных данных после выполнения программы получаются те же результаты. Поэтому для моделирования случайных процессов на ЭВМ формируют последовательность некоррелированных (не связанных между собой) чисел, называемых псевдослучайными. В частности, для моделирования случайных чисел с равномерным распределением в интервале (0, 1) преобразуют число x_i из этого интервала в несколько раз большее многозначное число, а затем отбрасывают целую часть результата, снова получая число x_{i+1} из того же интервала (0, 1).

При использовании микрокалькуляторов для подобного преобразования среди описанных в литературе достаточно удобна формула

$$x_i = e^{x_{i-1} + \pi} - E[e^{x_{i-1} + \pi}],$$

где E — символ целой части содержащего скобок.

Эта формула, реализуемая минимальным количеством операторов, обеспечивает формирование достаточно длинных для большинства практических задач последовательностей некоррелированных (псевдослучайных) чисел из интервала (0, 1).

Разбив интервал на две части в соотношении, соответствующем заданным вероятностям $p(1)$ и $p(0) = 1 - p(1)$, и организовав высвечивание символов 1 и 0 при попадании формируемого псевдослучайного числа в соответствующий интервал, можно составить модель случайного события с заданной вероятностью двух исходов в виде программы 26. Так как генерируемая последовательность исходов случайного события будет зависеть от выбора начального значения числа x_0 из интервала (0, 1), то для его случайного выбора составная программа 26 дополнена программой с оператором косвенного вызова из памяти на адресном регистре 0.

Программу 26 можно сократить, отбросив дополнительную программу для формирования x_0 и задавая это число в регистр 9 перед началом выполнений программы. В этом случае необходимо обеспечить случайный выбор x_0 , так как каждому из таких чисел будет соответствовать определенная последовательность формируемых символов исходов. Так, при $p(1) = p(0) = 0,5$ и $x_0 = 0,5$ будет формироваться последовательность символов исходов $y_i = 0; 1; 0; 0; 1; 1; 0; 0; 1; 1; \dots$

Случайное событие с k равновероятными исходами удобно формировать, разбив интервал (0, 1) на k равных частей и организовав высвечивание номера исхода при попадании квазислучайного числа из интервала (0, 1) в соответствующий частный интервал. Такой прием реализован в программе 27, которую при желании можно дополнить программой для формирования начального значения x_0 , содержащейся в программе 26, с изменением адресов переходов.

Начальное значение x_0 перед выполнением этой программы следует выбирать по возможности случайным. При моделировании по этой програм-

Программа 26. Моделирование случайного события с заданной вероятностью исходов $p(1)$ и $p(0) = 1 - p(1)$

ИП9 69	π 20	+	10	e^x 16	П9 49	КИП9 Г9	\rightarrow 25	ИП9 69	— 11	П9 49
ИП7 67	— 11	$x < 0$ 5C	17	Сх 0Г	БП 51	18 18	1 01	С/П 50	БП 51	
00 00	Сх 0Г	1 01	0 00	П0 40	$1/x$ 23	П9 49	КИП0 Г0	ИП0 60	$x = 0$ 5E	
25 25	БП 51	21 21								

Инструкция. Ввести $p(1) = P7$, нажать клавиши БП 2 1 С/П и через 5—10 с остановить программу нажатием клавиши С/П;

нажимать клавиши В/О и С/П (первый раз обязательно) или С/П для высвечивания символов 1 и 0 исходов с заданной вероятностью.

Программа 27. Моделирование случайного события с k равновероятными исходами $y_i \in \{1, 2, \dots, k\}$

ИП9 69	π 20	+	10	e^x 16	П9 49	КИП9 Г9	\rightarrow 25	ИП9 69	— 11	П9 49
ИП7 67	\times 12	1 01	+	10	П8 48	КИП8 Г8	ИП8 68	С/П 50	БП 51	00 00

Инструкция. ($k = P7$); $x_0 = P9$ (В/О)

С/П $PX = y_i$.

ме падения кубика вверх одной из граней (с вероятностью $1/6$ для каждой грани) для $6 = P7$ и $0,5 = P9$ получим $y_i = 1, 6, 1, 2, 4, 6, 3, 3, 6, 5, 5, 2, 4, 6, 1, 1, 4, 1, 4, 3, \dots$

Программу 27 можно использовать, в частности, для игры в спортлото или случайного выбора фишек в обычном лото, приняв $k = 100$. В этом случае после каждого выполнения программы с вероятностью 0,01 будет высвечиваться одно из чисел от 0 до 99. Например, при $k = 100$ и $x_0 = 0,5$ после шести выполнений программы получим номера $y_i = 16, 96, 9, 18, 55, 88$. Следует учитывать возможность повторения одинаковых номеров, так как для формирования используются только два первых разряда чисел из интервала (0, 1).

Программы, подобные рассмотренным, применимы в различных играх с полноправным участием микро-

ЭВМ. В большинстве таких игр микро-ЭВМ, программы работы которых обеспечивают практически случайный выбор решения с заданной вероятностью, имеют преимущества перед человеком и обычно оказываются более удачливыми игроками.

В качестве простейшего примера рассмотрим игру «чет — нечет», один из участников которой угадывает одно из двух возможных решений (выбор четного или нечетного числа, состояния некоторого объекта и т. п.). Если обозначить эти решения символами 1 и 0, то принимающий их участник игры будет иметь две чистые стратегии: принять $y_1 = 1$ или принять $y_2 = 0$. Отгадывающий решения участник игры также имеет две чистые стратегии: предсказать $x_1 = 1$ или предсказать $x_2 = 0$. Оценивая правильное предсказание баллом 1 и ошибочное — баллом — 1, получим для

предсказывающего участника матрицы игры с чистыми стратегиями, соответствующую двум первым строкам табл. 4.

Такая матрица с чистыми стратегиями не имеет седловой точки и, следовательно, выбор чистых стратегий нецелесообразен. Действительно, при выборе участником игры единственной стратегии противник, определив эту стратегию, выиграет. В связи с симметрией матрицы чистых стратегий для игры «чет — нечет» можно ожидать, что оптимальной является смесь с равными долями стратегий. Это подтверждается и вычислением по приведенным ранее формулам оптимальных долей стратегий $p_x = p_y = 0,5$, а также равных нулю элементов дополнительных строки и столбца, соответствующих оптимальной смеси стратегий (табл. 4). В этом случае максимин и минимакс равны нулю и чистая цена игры $\bar{C} = 0$.

При реализации оптимальной стратегии в игре с человеком существенна не только доля чистых стратегий, но и последовательность их изменения. Если один из игроков придерживается определенной закономерности в изменении стратегий, то второй игрок воспользуется этим в своих интересах. Поэтому выбор чистых стратегий в игре «чет — нечет» должен изменяться случайным образом с вероятностью $p = 0,5$ для каждой чистой стратегии. Если один из игроков придерживается этой стратегии, то независимо от стратегии второго участника при большом числе ходов игра заканчивается ничейным результатом.

Для игры программируемого микрокалькулятора в «чет — нечет» с оптимальной стратегией можно использовать программу 26 при $p(1) = 0,5$. Во избежание ошибки или соблазна скорректировать результаты

4. Матрица игры «чет — нечет»

	$y_1 = 1$	$y_2 = 0$	$y_3 = (y_1 + y_2)/2$
$x_1 = 1$	1	-1	0
$x_2 = 0$	-1	1	0
$x_3 = (x_1 + x_2)/2$	0	0	0

противнику следует записывать свои решения или предсказания на бумаге для сравнения с предсказаниями или решениями микрокалькулятора на том же ходе. При небольшом числе ходов можно выиграть у микрокалькулятора, но по мере увеличения этого числа результат будет все увереннее приближаться к ничейному.

В игре «чет — нечет» между людьми основное значение имеет знание психологии противника для выбора своей оптимальной стратегии. Алгоритм изучения психологии противника можно составить различными способами. Простейший из них основан на накоплении данных о числах n сохранений и m изменений стратегий противника в предыдущих ходах. Если противник более склонен к сохранению принятого ранее решения, то на очередном ходе целесообразно выбирать решение, совпадающее с решением противника на предыдущем ходе. В противном случае следует выбирать противоположное решение с алгоритмом, схема которого для загадывающего участника игры показана на рис. 27. Для отгадывающего игрока следует заменить в этой схеме условие $m \leq n$ условием $m \geq n$. На входном языке программируемых микрокалькуляторов этот алгоритм реализован программой 28, перед каждым выполнением которой противнику следует записать на бумаге свое предсказание или принятое решение.

Микрокалькулятор, «обученный» по программе 28, играет в «чет — нечет» подобно человеку, руководствуясь аналогичной стратегией и выигрывает у недостаточно проница-

тельного противника. Однако внимательный партнер, разгадав стратегию микрокалькулятора, будет выигрывать при выборе достаточно гибкой стратегии, не позволяющей микрокалькулятору «приспособиться» к психологии противника.

Так, при выборе принимающим решение противником казалось бы беспроигрышной стратегии с равными долями чистых стратегий в последовательности ходов

$$y_x = 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0$$

микрокалькулятор ответит последовательностью предсказаний

$$x_i = 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0$$

и выиграет со счетом 19 : 1.

В следующей партии с другой последовательностью выбранных решений без учета «психологии» микрокалькулятора, например

$$y_i = 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0$$

микрокалькулятор выбирает последовательность предсказаний

$$x_i = 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0$$

и выигрывает со счетом 13:7.

Однако при выборе стратегии со сложной закономерностью смены чистых стратегий противником микрокалькулятор может проиграть. Например, при последовательности решений

$$y_i = 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0$$

микрокалькулятор выбирает последовательность предсказаний

$$x_i = 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1$$

и проигрывает со счетом 7:13.

Подобным образом, хотя и с несколько большими трудностями, можно выиграть у микрокалькулятора и при выборе им решений (после замены в программе 28 команды $x \geq 0$ командой $x < 0$).

Для игры микро-ЭВМ с серьезным противником следует использовать алгоритм со статистической оценкой корреляционной связи [8, 9, 11] между несколькими предыдущими ходами противника. Простейший из таких алгоритмов с учетом связи между тремя последними ходами противника на входном языке микрокалькуляторов реализован программой 29.

Микрокалькулятор, «обученный» по программе 29, «размышляет» дольше, но играет более успешно, чем по предыдущей программе. Так при $a = 789$ и выборе противником после-

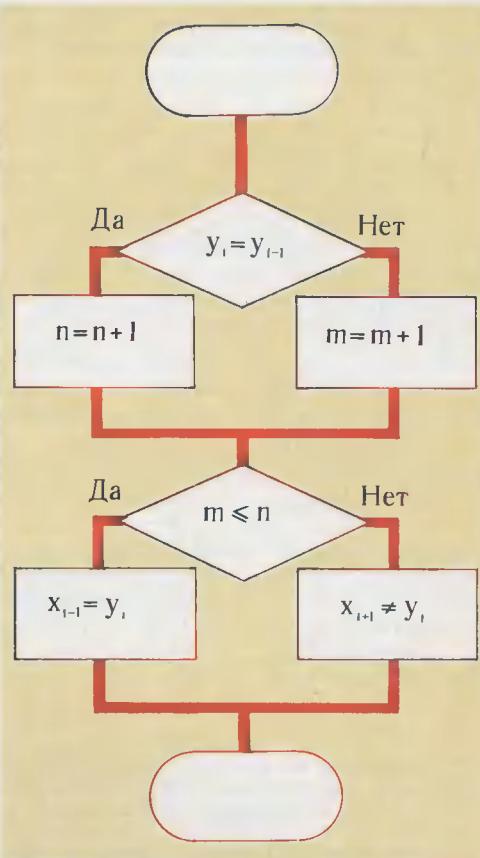


Рис. 27.

Программа 28. Игра «чет — нечет» с учетом предыдущих ходов

P8 48	ИП3 63	— 11	x < 0 5C	06 06	КИП6 Г6	ИП9 69	1 01	— 10	П9 49
ИП8 68	ИП7 67	ХY 14	П7 47	— 11	x = 0 5E	20 20	КИП4 Г4	БП 51	21 21
КИП5 Г5	ИП4 64	ИП5 65	— 11	x ≥ 0 59	29 29	ИП8 68	БП 51	35 35	ИП8 68
1 01	— 11	x ≠ 0 57	35 35	1 01	П3 43	С/П 50	БП 51	00 00	Сх 0Г
П4 44	П5 45	П6 46	П7 47	П9 49	БП 51	00 00			

Инструкция. Для принятия решений микрокалькулятором заменить в программе команду $x \geq 0$ по адресу 24 командой $x < 0$; при первом пуске программы нажать клавиши В/О и С/П или С/П; после высвечивания пред-

сказания (решения) 1 или 0 ввести в регистр X предыдущее решение (предсказание) противника; общее число ходов микрокалькулятора и ходов с его выигрышем хранятся соответственно в регистрах 9 и 6.

Программа 29. Игра «чет — нечет» с учетом корреляции ходов противника

П9 49	ИПА 6—	— 11	x = 0 5E	06 06	КИП4 Г4	ИП9 69	2 02	× 12	1 01
— 11	ПД 4Г	ИП3 63	Х 12	ПС 4С	КИП5 Г5	ИПД 6Г	ИП2 62	П3 43	Х 12
ИПД 6Г	ИП1 61	П2 42	Х 12	ИП6 66	+	10	П6 46	ИПД 6Г	П1 41
ХY 14	ИП7 67	+	10	П7 47	ИП2 62	Х 12	ИПС 6С	ИП8 68	+
ИП3 63	Х 12	+	10	+	10	x ≥ 0 59	49 49	1 01	БП 51
ПА 4—	С/П 50	БП 51	00 00	П9 49	9 09	П0 40	Сх 0Г	КП0 L0	ИП0 60
1 01	— 11	x = 0 5E	57 57	ИП9 ИП9	sin 1С	x ≥ 0 59	69 69	Сх 0Г	П9 49
БП 51	06 06								

Инструкция. При игре микрокалькулятора в качестве принимающего решения заменить в программе команду $x \geq 0$ по адресу 44 командой $x < 0$; установить переключатель Р — Г в положение Р; ввести трехзначное число a в регистр X и нажать клавиши БП54С/П (время счета около 35 с); после высвечивания

каждого предсказания (решения) 1 или 0 микрокалькулятора вводить в регистр X предыдущее решение (предсказание) противника и нажимать клавиши В/О и С/П или С/П (время счета около 15 с); числа всех ходов микрокалькулятора и его ходов с выигрышем хранятся соответственно в регистрах 5 и 4.

довательности решений

$$y_i = 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0$$

микрокалькулятор выигрывает со счетом 20:0, а при последовательности решений противника

$$y_i = 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0$$

микрокалькулятор формирует последовательность предсказаний

$$x_i = 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0$$

и выигрывает со счетом 18:2.

При вводе противником последовательности решений с менее явно выраженной закономерностью смены стратегий

$$y_i = 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1$$

микрокалькулятор проигрывает со счетом 9:11, так как последовательность относительно коротка — при ее повторении микрокалькулятор предсказывает 14 из 20 ходов противника и выигрывает с общим счетом 23:19.

Программы 28 и 29 не реализуют оптимальную беспрогрышную стратегию, но при достаточно большом числе ходов микрокалькулятор (в особенности, с программой 29) чаще всего выигрывает у человека. Еще успешнее подобные программы можно использовать в играх с природой, не пытающейся разгадать стратегию микро-ЭВМ. Простейшим примером является предсказание погоды на следующий день в двоичной оценке (хорошая погода — плохая погода, будет дождь — не будет дождя и т. п.) по данным за предыдущие дни. Так как влияние циклонов и антициклонов проявляется в достаточно длинных периодах плохой или хорошей погоды, то программы [11], подобные программам 28 или 29, обеспечивают достаточно уверенное предсказание погоды на следующий день.

Возможность выбора с помощью микро-ЭВМ случайного решения с тре-

буемой вероятностью особенно эффективно проявляется в играх с оптимальной выигрышной стратегией участника, в роли которого выступает машина. Это, в частности, относится к группе игр с неполной информацией, которым можно дать общее название «верю — не верю».

Первый участник таких игр детерминированно (по определенному правилу) или случайно выбирает исход 0 или 1 некоторого события, но его высказывание о выбранном исходе, сообщаемое второму участнику игры, может быть истинным или ложным. Противник отвечает «верю» или «не верю», и выигрыши или проигрыши участников в зависимости от сочетаний выбранных ими стратегий оцениваются в баллах по условиям игры.

Следовательно, первый участник такой игры располагает четырьмя чистыми стратегиями, тогда как его противник — только двумя. Рассмотрим вариант игры с детерминированным выбором исхода события, например «записал 1» или «записал 0», и матрицей выигрыша первого участника в баллах на каждом ходу, показанной в табл. 5.

Первый игрок практически использует лишь первые две стратегии, так как стратегия x_3 проигрышна при любой стратегии противника, а стратегия x_4 чрезмерно рискована, кроме

5. Матрица игры «верю — не верю» с детерминированным выбором исхода события первым участником

	$y_1 : \text{«верю»}$	$y_2 : \text{«не верю»}$
$x_1 : \text{записал } 1, \text{ сказал } \text{«записал } 1\text{»}$	0	2
$x_2 : \text{записал } 0, \text{ сказал } \text{«записал } 1\text{»}$	1	-2
$x_3 : \text{записал } 1, \text{ сказал } \text{«записал } 0\text{»}$	-1	-2
$x_4 : \text{записал } 0, \text{ сказал } \text{«записал } 0\text{»}$	-1	2

$x = 0$	5Е	15	ИП8 68	$x = 0$	10	ИП7 67	2	—	БП 51	21	21
ИП7 67	2	02	+	10	БП 51	21	ИП8 68	$x \neq 0$ 57	22	ИП7 67	1 01
+	10	П7 47	КИП4 Г4	ИП7 67	С/П 50	ИП9 69	π	20	+	e ^x 16	П9 49
КИП9 Г9	→	25	ИП9 69	—	П9 49	0 00	,	0—	6 06	—	$x < 0$ 5С
44	44	1 01	БП 51	45 45	Сх 0Г	П8 48	1 01	С/П 50	БП 51	00 00	
П9 49	Сх 0Г	П4 44	П7 47	БП 51	25 25						

Инструкция. Ввести в регистр X положительное число $a < 10$ и нажать клавиши БП50С/П; после каждого высвечивания 1 (высказывание «записал 1») сохранить содержимое регистра X при ответе «верю» или стереть его клавишей Сх при ответе «не верю» и нажать только клавишу С/П, после чего

высвечивается суммарный выигрыш микрокалькулятора (он также хранится в регистре 7, а число ходов микрокалькулятора в регистре 4); проверив при желании истинное решение 1 или 0, хранящееся в регистре 8, нажать клавиши В/О и С/П или С/П для формирования следующего высказывания «записал 1».

того, при стратегиях x_3 и x_4 (утверждение «записал 0») второй участник выигрывает при выборе стратегии y_1 («верю»). Следовательно, для первого участника игра сводится к матрице 2×2 , образованной двумя первыми строками табл. 5. Эта матрица не содержит седловой точки и поэтому оптимальными для обоих игроков являются смешанные стратегии.

По приведенным ранее формулам находим оптимальную смесь стратегий первого участника игры с долями $p_{x1} = (a_{11} - a_{21}) / (a_{11} + a_{22} - a_{12} - a_{21}) = 0,6$ и $p_{x2} = 1 - p_{x1} = 0,4$ чистых стратегий. Аналогично находим оптимальные доли чистых стратегий второго участника $p_{y1} = 0,8$ и $p_{y2} = 0,2$. Цена игры $\bar{D} = a_{11}p_{x1} + a_{12}p_{x2} = 0,4$, и при большом числе ходов первый участник при оптимальной смеси стратегий в среднем выигрывает 0,4 балла на каждом ходу.

Таким образом, первому участнику игры выгодно лишь высказывание

«записал 1» независимо от истинности этого утверждения при случайному выборе истинного решения «записал 1» с вероятностью $p(1) = 0,6$ и истинного решения «записал 0» с вероятностью $p(0) = 1 - p(1) = 0,4$. Следовательно, составление программы для микро-ЭВМ в качестве первого участника игры в основном сводится к организации случайного выбора истинного решения в оптимальной смеси чистых стратегий. Эта задача на входном языке микрокалькуляторов решена с помощью программы 30, дополненной фрагментом для оценки выигрыш согласно матрице игры и автоматизации вспомогательных операций по очистке регистров памяти.

Пусть принято $a = 9$ (это число целесообразно изменять перед каждой игрой) и микрокалькулятор высветил 1 («записал 1»). Поверив высказыванию, нажмем клавишу С/П и получим выигрыш микрокалькуля-

тора $PX = 1$. Проверка содержимого регистра 8 подтверждает, что высказывание микрокалькулятора было ложным и он выиграл 1 балл. Нажав клавишу С/П и получив $PX = 1$ («записал 1»), на этот раз нажмем клавиши Сх («не верю») и С/П, что приведет к высвечиванию общего выигрыша микрокалькулятора $PX = 3$. Продолжая игру, нетрудно убедиться, что при достаточно большом числе ходов микрокалькулятор неизбежно выигрывает.

Согласно табл. 5 для второго участника стратегия «верю» проигрышна и, казалось бы, целесообразно использовать лишь стратегию «не верю». Однако она более рискована и минимальный проигрыш второго участника соответствует его оптимальной смеси стратегий с долей $p_{y1} = 0,8$ стратегии «верю» и $p_{y2} = 0,2$ стратегии «не верю».

В некоторых вариантах рассматриваемой игры между людьми первый участник выбирает исход события случайным образом с вероятностью двух возможных исходов $p(1) = p(0) = 0,5$, а матрица игры также имеет размер 2×2 . Рассмотрим подобный вариант со следующими правилами: первый участник вынимает из тасованной колоды карточек (половина которых обозначена нулем и половина единицей) одну и, не показывая ее противнику, утверждает «вытянул 1» или «вытянул 0». Выигрыши первого участника в зависимости от ответов второго «верю» или «не верю» указаны в табл. 6, где для сочетания стратегий x_4 и x_2 указан выигрыш — 1 вместо симметричного — 2, так как в последнем случае стратегия первого участника x_2 становится проигрышной и теряет смысл, а утверждение «вытянул 1» будет явно соответствовать стратегии x_1 .

Определим оптимальную стратегию первого участника в этом варианте

6. Матрица игры «верю — не верю» при случайном выборе исхода события первым участником

	y_1 : «верю»	y_2 : «не верю»
x_1 : вытянув 1, сказать «вытянул 1»	1	2
x_2 : вытянув 1, сказать «вытянул 0»	-1	-2
x_3 : вытянув 0, сказать «вытянул 1»	1	-2
x_4 : вытянув 0, сказать «вытянул 0»	-1	-1

7. Матрица игры «верю — не верю» с комбинированными стратегиями первого участника

	y_1 : «верю»	y_2 : «не верю»
x'_1 : сказать «вытянул 1»	1	0
x'_2 : сказать правду	0	0,5

игры. Если участник вытянул 1, ему достаточно стратегии x_1 , так как в этом случае он выигрывает при любом ответе противника. Если же он вытянул 0, то ему требуется две стратегии x_3 и x_4 . Поэтому для первого участника игры имеют смысл только две стратегии: сказать «вытянул 1» независимо от истинного исхода случайного события (что соответствует стратегиям x_1 и x_2 в табл. 6) и стратегия «сказать правду» или смесь стратегий x_1 и x_4 из табл. 6.

С учетом вероятности исходов случайного события (вынимания карточки из колоды) элементы матрицы игры с двумя комбинированными стратегиями первого участника $a'_{11} = a_{11}p(1) + a_{31}p(0) = 1$; $a'_{12} = a_{12}p(1) + a_{32}p(0) = 0$; $a'_{21} = a_{11}p(1) + a_{41}p(0) = 0$ и $a'_{22} = a_{12}p(1) + a_{42}p(0) = 0,5$ (табл. 7).

Составленная матрица игры с комбинированными стратегиями первого участника не содержит седловой точки, и оптимальная смесь чистых стратегий x'_1 и x'_2 с долями $p'_{x1} = 0,5/(1 + 0,5) = 1/3$ и $p'_{x2} = 1 - p'_{x1} = 2/3$. Це-

$x \neq 0$	15	15	ИП6 66	$x = 0$	10	10	ИП7 67	1	01	—	11	БП 51	29	29
ИП7 67	1	01	+	10	БП 51	30	30	ИП8 68	$x \neq 0$	23	23	ИП7 67	2	02
+	10	БП 51	29	29	ИП6 66	$x \neq 0$	57	05	05	ИП7 67	2	02	—	11
КИП4 Г4	ХY 14	С/П 50	ПП 53	60	60	2	02	1/x	23	—	11	$x < 0$	43	43
1	01	БП 51	44	44	Сх 0Г	П8 48	ПП 53	60	60	3	03	1/x	23	—
$x < 0$	55	1	БП 51	56	56	ИП8 68	П6 46	С/П 50	БП 51	00	00			
ИП9 69	π 20	+	10	e^x	16	П9 49	КИП9 Г9	→	25	ИП9 69	—	11	П9 49	
В/О 52	П9 49	Сх 0Г	П4 44	П7 47	БП 51	33	33							

Инструкция. Ввести в регистр X положительное число $a < 10$ и нажать клавиши БП71С/П, после высвечивания 1 («вытянул 1») или 0 («вытянул 0») сохранить в регистре X или ввести 1 («верю») или 0 («не верю») и нажать только клавишу С/П для высвечивания суммарного выигрыша микрокалькулятора.

калькулятора (число его ходов хранится в регистре 4) и, при желании, проверки истинного исхода 0 или 1, хранящегося в регистре 6: нажать клавиши В/О и С/П или С/П для формирования следующего высказывания микрокалькулятора.

на игры $Q = 1/3$ и при достаточно большом числе ходов первый участник в среднем выигрывает $1/3$ балла за один ход. Так как целесообразна случайная смесь чистых стратегий в установленной пропорции, то их доли равны вероятностям случайного выбора комбинированных стратегий первым участником.

Для второго участника оптимальная смесь чистых стратегий $p'_{y1} = (a'_1 - a'_2)/(a'_1 + a'_2 - a'_1 - a'_2) = 2/3$ и $p'_{y2} = 1/3$. Следовательно, его минимальный проигрыш в игре с человеком утверждений «верю» или «не верю» с вероятностями $2/3$ и $1/3$. В то же время при игре с участием микро-ЭВМ, не учитывающей случайность выбора ходов второго игрока, оптимальная стратегия последнего

соответствует, например, высказыванию «не верю» через каждые два высказывания «верю».

При участии микро-ЭВМ в качестве первого игрока необходимо моделировать случайный выбор двух возможных исходов события (выбора карточки с символами 0 или 1) с вероятностями 0,5 и случайный выбор комбинированных стратегий x'_1 и x'_2 с вероятностями $1/3$ и $2/3$. При составлении программы на входном языке микрокалькулятора следует предусмотреть выполнение фрагмента, подобного программе 26, для случайного выбора исходов 1 и 0 с вероятностью 0,5 и повторное выполнение этого же фрагмента с выбором высвечиваемого символа в соответствии с оптимальной комбинированной стратегией. Эти операции пре-

дусмотрены в программе 31, дополненной фрагментами для вычисления выигрыша микрокалькулятора согласно исходной матрице (табл. 6), вычисления числа ходов и вспомогательных операций.

Выбрав $a = 5$ и пустив программу 31 нажатием клавиш БП 7 1 С/П, получим $PX = 1$ («вытянул 1»). Сохранив содержимое регистра X , нажмем клавишу С/П для определения выигрыша микрокалькулятора $PX = 1$ и снова нажмем клавишу С/П для высвечивания следующего хода микрокалькулятора $PX = 0$ («вытянул 0»). Сохранив это содержимое

регистра X и снова поверив микрокалькулятору, нажмем клавишу С/П, что приведет к результату $PX = 0$, что означает проигрыш микрокалькулятора на этом ходе. Однако, продолжая игру, легко убедиться, что микрокалькулятор выигрывает тем увереннее, чем больше ходов в партии. Рассмотренные методы выбора оптимальных смесей чистых стратегий можно использовать для составления алгоритмов и программ участия микро-ЭВМ и в различных других играх с неполной информацией.

Глава 6

МИКРО- ЭВМ- СУДЬЯ

Успех участников многих игр зависит не только от умения логически мыслить, но еще в большей степени от глазомера, быстроты реакции и необходимых для победы навыков. Это относится, в частности, к спортивным и военно-патриотическим играм, а также к их различным моделям, реализуемым игровыми автоматами. Микро-ЭВМ могут успешно участвовать в подобных играх в качестве члена судейской коллегии, измеряющего нарушения правил, строгого экзаменатора, оценивающего знания состязающихся, и даже квалифицированного тренера-наставника, обучавшего своих учеников.

Спортивное судейство прежде всего связано с измерением отрезков времени. Цифровые ЭВМ, которые могут быть привлечены к выполнению судейских обязанностей, содержат генератор тактовой частоты (иногда называемый часами) для определения с высокой точностью последовательности одинаковых интервалов времени (тактов). Это обеспечивает возможность вывода на дисплей персональных ЭВМ цифровых отсчетов времени, а некоторые из таких машин снабжены встроенной программой для вывода на дисплей изображения часов с движущимися стрелками.

Программируемые микрокалькуляторы также содержат генераторы тактовой частоты, определяющей время выполнения каждого оператора входного языка. При этом в связи с технологическим разбросом параметров тактовые частоты микрокаль-

куляторов даже одного типа несколько отличаются и в некоторой степени зависят от температуры окружающей среды и напряжения источника питания. Это обстоятельство приходится учитывать при составлении программы для отсчета времени и корректировать исходные данные (как и при регулировке хода обычных часов) программы для используемого микрокалькулятора.

Формирование непрерывных отсчетов времени с помощью микрокалькулятора обеспечивается программой без оператора остановки С/П, содержащей замкнутый цикл. Так как время каждого выполнения такого цикла практически постоянно, то достаточно формировать цифровые отсчеты числа выполненных циклов в масштабе выбранной единицы измерения времени.

Примером может служить программа Сx ПП₁ КИПП₁ ИПП₁ ИПП₂ × × ПП₃ БП 02 с оператором косвенного вызова из памяти на адресном регистре $N_1 = 4,5$ или 6. При каждом выполнении такого цикла, включающего семь последних операторов программы, содержимое регистра N_1 , равное числу выполненных итераций, умножается на экспериментально найденный масштабный множитель k , содержащийся в регистре N_2 , и в регистре N_3 заносится текущее время выполнения программы в выбранных единицах измерения, изменяющееся после каждого цикла на величину k .

Для индикации формируемых отсчетов времени удобно использо-

Программа 32. Цифровые часы

Сх 0Г	6 06	0 00	П7 47	1 01	0 00	0 00	П8 48	КИП5 Г5	ИП7 67
ИП5 65	5 05	÷ 13	П6 46	— 11	x < 0 5C	20 20	КИП4 Г4	Сх 0Г	ИП5 45
ИП6 66	ИП8 68	÷ 13	ИП4 64	+	10	↑ 0Е	↑ 0Е	↑ 0Е	↑ 0Е
↑ 0Е	↑ 0Е	↑ 0Е	↑ 0Е	↑ 0Е	↑ 0Е	↑ 0Е	↑ 0Е	↑ 0Е	↑ 0Е
↑ 0Е	↑ 0Е	↑ 0Е	↑ 0Е	↑ 0Е	БП 51	08 08			

Инструкция. (Для начальной установки часов ввести 22 = $P_4,0 = P_5$ и, нажав клавиши В/О и С/П, определить время формирования цифры 1 в высвечиваемом числе 22,01 — если оно отличается от 60 с, то соответственно увеличить или уменьшить число операторов ↑ в программе); ввести начальные

целое число часов $n = P_4$ и число минут $m = = P_5$ и нажать клавиши В/О и С/П; при выполнении операторов ↑ на индикаторе высвечиваются (в левых знакоместах) текущие целое число $P_4 = n_i$ часов и целое или дробное (с точностью до 0,2) число $P_5 = m_i$ минут.

вать имеющийся во входных языках некоторых программируемых микрокалькуляторов оператор PAUSE для высвечивания на несколько секунд результата предыдущих операций в процессе выполнения программы. Во входном языке микрокалькуляторов используемых нами типов такой оператор отсутствует, и для лучшего выделения на индикаторе текущего отсчета времени следует обеспечить его хранение в регистре X в течение достаточно большой части времени выполнения цикла. Этим условиям отвечает программа 32, обеспечивающая вывод отсчетов времени в часах и минутах на индикатор в процессе выполнения программы.

Программа 32 по удобству отсчета времени не может соперничать с обычными часами и представляет интерес лишь как пример моделирования устройства для отсчета часов и минут. Однако для некоторых игр (например, при учете времени, затрачиваемого шахматистами на обдумывание ходов) обычные часы непригодны, и моделирование шахматных

часов может оказаться целесообразным.

В этом случае достаточно отсчитывать время в минутах, для чего целесообразно также использовать цикл с оператором косвенного вызова из памяти на адресном регистре с номером $N = 4, 5$ или 6 , но для коррекции масштаба времени следует предусмотреть запись масштабного множителя в регистр памяти. Суммирование времени обдумывания ходов шахматистами должно реализоваться в различных регистрах памяти, что можно обеспечить установкой ($f \neq 0$) и снятием ($f = 0$) флага с проверкой его условным оператором. Этим условиям отвечает программа 33, в которой действие флагка шахматных часов моделируется автоматической остановкой программы с высвечиванием числа минут (с отрицательным знаком), превышающим допустимое суммарное время обдумывания игроком, сделавшим предыдущий ход.

Для упрощения программы 33 перед ее выполнением задается предельное допустимое время обдумывания

Программа 33. Цифровые шахматные часы

ИП6 66	$x = 0$ 5Е	29 29	ИП5 65	П6 46	ИП8 68	+	П8 48	ИП0 60	ХУ 14
— 11	$x < 0$ 5С	20 20	С/П 50	Сх 0Г	П5 45	П6 46	П7 47	П8 48	В/О 52
Сх 0Г	П4 44	КИП4 Г4	ИП4 64	ИП9 69	×	П5 45	БП 51	22 22	Сх 0Г
П6 46	ИП5 65	ИП7 67	+	10	П7 47	БП 51	08 08		

Инструкция. (Для начальной регулировки часов ввести $k = 0,25 = P9$, нажать клавиши БП20С/П и через 60 с остановить программу нажатием клавиши ПП — если содержимое регистра 5 больше или меньше единицы, то соответственно уменьшить или увеличить k), ввести допустимое время обдумывания ходов в регистр 0; для первого пуска программы нажать клавиши БП14С/П; для последующих — клавиши В/О и С/П, после

очередной остановки программы нажатием клавиш ПП или С/П в регистрах 7 и 8 хранятся отсчеты суммарного времени шахматистов без отсчета времени последнего хода, хранящегося в регистре 5; при автоматической остановке программы высвечивается число минут (с отрицательным знаком), превышающее допустимое время обдумывания ходов шахматистом, выполнившим предыдущий ход.

Программа 34. Цифровой секундомер

ИП9 69	\uparrow 0Е	\uparrow 0Е	\uparrow 0Е	С/П 50	+	10	БП 51	05 05
-----------	------------------	------------------	------------------	-----------	---	----	----------	----------

Инструкция. (Для начальной регулировки хода ввести $0,5 = P9$ и нажать клавиши В/О и С/П, что приведет к высвечиванию $PX = 0,5$; нажав клавишу С/П, через 60 с остановить программу — если высвечиваемые цифры отличаются от 60, то соответственно уточнить содержимое регистра 9), после ввода

подобранныго масштабного коэффициента в регистр 9 нажать клавиши В/О и С/П; в нужный момент времени пустить программу нажатием клавиши С/П и остановить программу после окончания процесса, длительность которого измеряется, что приведет к высвечиванию числа секунд.

независимо от числа ходов. Для того чтобы учитывалось допустимое время каждого хода, эту программу следует несколько усложнить, дополнив ее счетчиком шагов на операторе косвенной адресации и фрагментом вычисления допустимого времени обдумывания ходов в зависимости от их числа.

Относительно просто с помощью программируемого микрокалькулятора реализуется отсчет времени в секундах при остановке программы (как и в обычном секундомере). Если при этом время отсчитывается в секундах, то длительность выполнения цикла в программе без оператора

С/П должна быть меньше секунды. Этим требованиям отвечает программа 34, в которой после выполнения начального фрагмента для засылки масштабных коэффициентов в регистры операционного стека (что соответствует нажатию кнопки секундомера для сброса предыдущего отсчета) и нажатия клавиши С/П выполняется цикл суммирования в регистре Х масштабных слагаемых.

Во многих случаях для определения заданных интервалов времени удобно использовать будильник или, для небольших отрезков времени, таймер. Для моделирования таймера достаточно составить программу, содержа-

Программа 35. Цифровой таймер

ИП7 47	ИП9 69	\times	12	ПО 40	ИП7 67	С/П 50	L0	5Г	06 06	ИП7 67	С/П 50
-----------	-----------	----------	----	----------	-----------	-----------	----	----	----------	-----------	-----------

Инструкция. (Для корректировки хода таймера ввести корректирующий множитель $k = 2,3$ в регистр 9 и 60 в регистр X , нажать клавиши В/О и С/П и в фиксируемый момент времени клавишу С/П, если время выполнения программы меньше или больше 60 с, то соответственно увеличить или уменьшить содержа-

щую регистра 9), ввести $k = P9$, для каждого отсчета времени вводить заданное время в секундах в регистр X и нажимать клавиши В/О и С/П для начальной установки; в нужный момент нажать клавишу С/П, что приведет к высвечиванию заданного числа через равное число секунд.

щую цикл с заданным числом итераций, равным целой части произведения заданного числа секунд на масштабный множитель, определяемый временем выполнения одного цикла. Примером может служить программа-таймер 35 с оператором цикла L0 06 на адресном регистре 0.

При выполнении программы 35 ко времени выполнения циклов добавляется время выполнения операторов ИП7 и С/П (около 0,6 с). Поэтому для точного отсчета времени с погрешностью менее 0,6 с следует корректировать масштабный множитель для различного числа секунд. Однако в этом нет необходимости, так как время реакции человека при нажатии клавиш соизмеримо с возможной погрешностью корректировки.

Программируемый микрокалькулятор, «обученный» по программе 35, удобно использовать, в частности, для определения времени выдержки при печатании фотоснимков, так как относительно слабое свечение индикатора не засвечивает фотобумагу, а пользование обычными часами в затемненной фотолаборатории затруднительно.

Спортивные и другие игры, связанные с перемещением во времени и пространстве спортсменов или управляемых ими объектов, успешно моделируются с помощью персональных ЭВМ при отображении на дисплее изменяющейся игровой ситуации. При

моделировании таких игр с помощью программируемых микрокалькуляторов, выводящих лишь цифровую информацию, приходится использовать различные схемы и карты для отображения условий игры и, в необходимых случаях, изменений игровой ситуации.

Успех участников многих игр судьи определяют по затратам времени на выполнение определенных действий. При моделировании таких игр с помощью персональных ЭВМ, снабженных специальными приспособлениями для управления перемещением объектов на дисплее, возможен непрерывный отсчет времени для оценки результатов участников игры. При использовании программируемых микрокалькуляторов можно лишь косвенно оценивать затраты времени участниками состязания в зависимости от выбора ими определенных решений, отображаемых количественно.

В качестве простейшего примера такого моделирования, который может быть усложнен и представлен в другом внешнем оформлении, рассмотрим игру под названием «смешанный марафон». Участники этого спортивного состязания должны за кратчайшее суммарное время проехать из пункта C (рис. 28) на велосипеде со скоростью V_b до выбранной ими точки x , переплыть со скоростью V_p реку со скоростью течения V_r и добежать до точки финиша Φ со скоростью V_6 . Если расстояние между

Программа 36. Игра «смешанный марафон»

П7 47	ИПА 6—	ИПВ 6L	ИПЗ 63	÷ 13	П9 49	ИП0 60	× 12	— 11	ИП7 67
— 11	x^2 22	ИПС 6C	x^2 22	+ 10	$\sqrt{ }$ 21	ИП2 62	÷ 13	ИП9 69	+ 10
ИП7 67	ИП1 61	÷ 13	+ 10	6 06	0 00	×	1 01	ВП 0C	5 05
+	Bx 10	— 0	— 11	C/P 50	БП 51	00 00			

Инструкция. Перед игрой ввести выбранные исходные данные $V_p = P0$; $V_b = P1$; $V_n = P2$; $V_6 = P3$; $a = PA$; $b = PB$; $c = PC$ (скорости, км/ч; расстояния, км); после ввода

в регистр X выбранного очередным участником расстояния x нажать клавишу С/П (первый раз В/О и С/П) для вычисления времени участника в минутах с четырьмя знаками.

Программа 37. Игра «гонки с препятствиями»

ИПА 6—	sin 1C	ИПД 6Г	× 12	КИПВ GL	+	10	КПВ LL	ИПА 6—	cos 1Г	ИПД 6Г
×	12	ИПВ 6L	1 01	+ 10	ПВ 4L	→ 25	КИПВ GL	+	10	КПВ LL
ПА 6—	ИПВ 6L	1 01	+ 10	ПВ 4L	ИПС 6C	— 11	$x = 0$ 5E	30 30	ПВ 4L	С/П 50
БП 51	00 00	ПА 4—	ИПС 6C	ПО 40	Сх 0Г	КПО L0	ИП0 60	$x = 0$ 5E	35 35	
ПВ 4L	БП 51	00 00								

Инструкция. Перед началом игры установить переключатель Р — Г в положение Г, ввести число ($n \leq 5$) участников $2n = PC$, длину пробега $d = PD$; ориентируясь по карте с точкой старта в начале координат ($x_0 = y_0 = 0$), участники игры по очереди вводят в регистр угол φ направления движения и нажимают клавишу С/П (для первого пуска программы —

клавиши БП 3 2 С/П), что приводит к вычислению текущих координат его положения на карте $PX = x$; $PY = y$; при пересечении границы препятствия участник игры обязан на следующем ходу ввести предыдущее значение φ с обратным знаком, чтобы устранить нарушение.

точками С и Ф вдоль реки равно a , ширина реки b и расстояние от реки до финиша c , то общее время движения

$$t = x/V_b + b/V_n + \sqrt{(a - x - bV_p/V_n)^2 + c^2/V_6} .$$

Предполагая, что спортивная подготовка и, следовательно, скорости участников марафона одинаковы, победителем марафона (по лучшему

результату из трех попыток) окажется спортсмен, наиболее удачно выбравший расстояние x . Судить эти состязания можно поручить любой микроЭВМ и, в частности, микрокалькулятору, «обученному» по программе 36 для вычислений суммарного времени участников по составленной формуле.

Пусть $V_p = 4$ км/ч; $V_b = 20$ км/ч; $V_n = 3$ км/ч; $V_6 = 10$ км/ч; $a = 2$ км; $b = 0,2$ км; $c = 0,5$ км. Тогда при

выборе расстояния $x = 1; 1,5$ и $1,8$ км по программе 36 соответственно получим время марафона $t = 25,14; 18,76$ и $16,88$ мин. Для ограничения результата четырьмя знаками в этой программе использован фрагмент 1 ВП 5 + Вх—.

Условия игры «смешанный марафон» (и, соответственно, программы-судью) можно разнообразить, увеличивая число и вид этапов или количество решений, принимаемых участниками. Однако в этой игре недостаточно явно проявляется спортивный характер подобных состязаний, участники которых в ходе игры непрерывно оценивают свои и чужие успехи и неудачи с целью первыми достичь финиша и только после этого узнать суммарное время игры.

Непрерывное изменение игровой ситуации можно моделировать с помощью микро-ЭВМ, заменив его изменением этой ситуации через определенные промежутки времени Δt . Тогда каждое выполнение программы можно сопоставить с концом этого отрезка времени и организовать определение программой положения участников или управляемых ими

объектов через каждый отрезок времени Δt .

Положение участников или управляемых объектов обычно определяют в прямоугольной системе координат x и y , но для выбора перемещения за время Δt участникам удобно задавать угол φ между направлением движения и горизонтальной осью x и расстояние перемещения d . В этом случае для определения перемещения объекта из точки с координатами

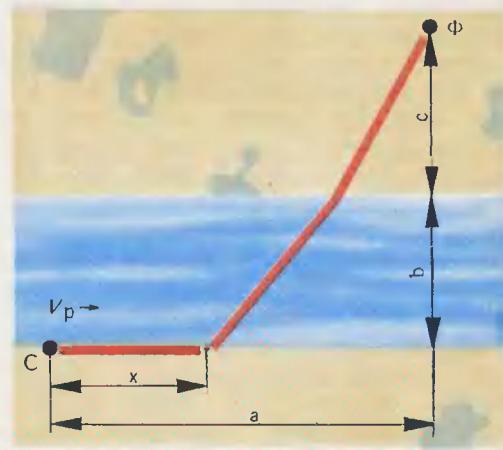


Рис. 28.

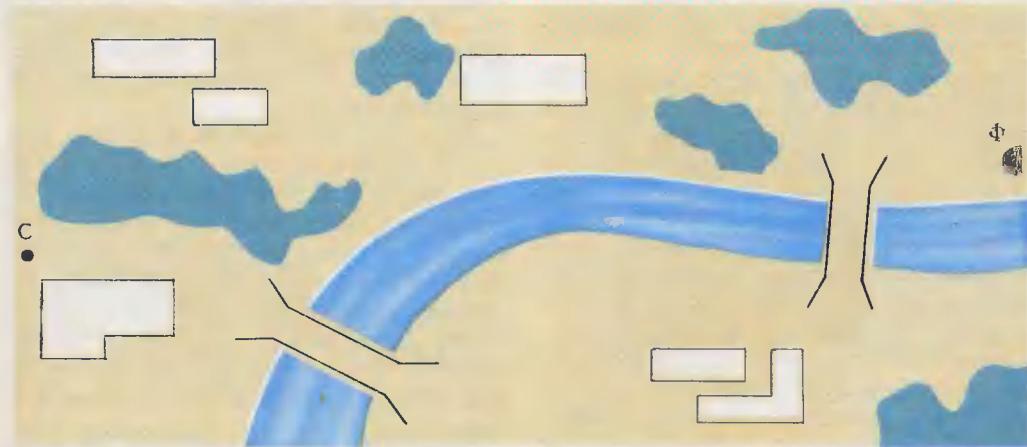


Рис. 29.

x_{i-1} и y_{i-1} в точку с координатами x_i и y_i достаточно вычислить $x_i = x_{i-1} + d_i \cos \varphi_i$, $y_i = y_{i-1} + d_i \sin \varphi_i$.

В простейшем случае расстояние d , на которое перемещается каждый объект за время Δt , можно принять постоянным. Тогда каждому участнику игры достаточно изменять или сохранять угол φ направления движения (с постоянной скоростью $d/\Delta t$) на каждом ходе, поручив микро-ЭВМ вычисление координат движущегося объекта через отрезки времени Δt . Очевидно, подобные игры имеют смысл лишь в том случае, когда их участники могут выбирать различные пути и заранее неизвестно, какой из них окажется короче и приведет к победе. Поэтому для такой игры на дисплее персональной ЭВМ или на бумаге должна быть изображена карта местности (например, показанная на рис. 29) с прямоугольной сеткой координат, указанием места старта (лучше в начале координат) и финиша (в простейшем случае в виде кружка радиусом $d/2$), а также границ препятствий, которые не должны нарушать участники игры. Правила такой «гонки с препятствиями» должны предусматривать наказания (например, пропуск хода виновного) для соперников, пересекших границы препятствий.

В рассматриваемой игре микро-ЭВМ должна хранить в памяти текущие координаты всех участников игры. В связи с простотой вычислений число участников ограничивается лишь емкостью числовой памяти и при использовании для этой игры в качестве технического эксперта программируемого микрокалькулятора, «обученного» по программе 37, в ней может участвовать до пяти соперников, координаты которых заносятся в 10 или менее регистров памяти операторами косвенного обращения к памяти.

Программа 37 достаточно универсальна и может использоваться с любой картой местности при выборе точки старта в начале координат. Уровень судейства машиной подобных состязаний повышается при автоматизации контроля за нарушениями правил игры и наказания нарушителей. Для этого достаточно выбирать границы препятствий, которые запрещается нарушать, так, чтобы они описывались достаточно простыми функциями, вычисляемыми машиной при контроле нарушений (рис. 30).

Для примера выберем круговую гоночную дорожку, ограниченную окружностями с радиусами R_{\min} и R_{\max} с точкой старта в средине дорожки на расстоянии $R = (R_{\min} + R_{\max})/2$ от центра окружностей. Совместив точку старта с началом координат, получим неравенство $R_{\max} > (x_i - R)^2 + y_i^2 > R_{\min}$, которому удовлетворяют координаты всех точек на гоночной дорожке.

При продвижении на каждом ходу на расстояние d участники игры выбирают угол φ между направлением движения и положительной осью координаты x . Если при этом координаты новой точки не удовлетворяют неравенству, то участник игры выехал за пределы гоночной дорожки и штрафуется машиной, сохраняющей координаты его положения перед очередным ходом.

На входном языке микрокалькуляторов подобные гонки двух участников контролируются программой 38, в которой вычисляются текущие координаты обоих участников и, если они находятся в пределах гоночной дорожки, заносятся в регистры памяти. Для тренировок можно выбрать сравнительно широкую дорожку, например, с $d = 2$; $R = 10$; $R_{\min} = 8$; $R_{\max} = 12$.

Точность моделирования спортивных состязаний повышается при выбо-

Программа 38. Игра «гонки по круговому треку»

П9 49	ИП7 67	ИП8 68	ПП 53	22 22	П8 48	ХY 14	П7 47	2 02	С/П 50
П9 49	ИП4 64	ИП5 65	ПП 53	22 22	П5 45	ХY 14	П4 44	1 01	С/П 50
БП 51	00 00	ИП9 69	sin 1C	ИПА 6—	Х 12	П3 43	+	10	П2 42
ХY 14	ИП9 69	cos 1Г	ИПА 6—	Х 12	П0 40	+	10	П1 41	ИПВ 6L
х ² 22	+	10	П6 46	ИПС 6C	— 11	х ≥ 0 59	55 55	ИП6 66	ИПД 6Г
х < 0 5C	55 55	ИП1 61	ИП2 62	В/О 52	ИП1 61	ИП0 60	— 11	ИП2 62	ИП3 63
— 11	В/О 52	Сх 0Г	П4 44	П5 45	П7 47	П8 48	БП 51	18 18	

Инструкция. Установить переключатель Р — Г в положение Г, ввести $d = PA$; $R = PB$; $R_{\min}^2 = PC$; $R_{\max}^2 = PD$ и нажать клавиши БП 6 2 С/П; после каждого выполнения программы высвечивается номер участника гонок, который вводит в регистр X выбранное

ре участниками игры не только угла φ направления движения, но и расстояния пробега d или скорости движения $d/\Delta t$ на каждом шаге. В простейшем случае для этого можно модифицировать программу, подобную программе 37, дополнив ее операторами для запоминания вводимых значений d . Однако целесообразно усложнить условия игры, вводя дополнительные условия.

В качестве примера рассмотрим моделирование авторалли на местности, карта которой показана на рис. 31. Участники должны проехать от места старта любым путем в город Ф, огибая заштрихованные озера и при необходимости изменяя на каждом отрезке времени Δt , соответствующем выполнению программы для каждого участника, поворот автомобиля для движения под углом φ к горизонтальной оси координат и удельную силу тяги $P \leq 1$, пропорциональную ско-

значение угла φ направления движения и нажимает только клавишу С/П; текущие координаты участников хранятся в регистрах $P7 = x'_i$; $P8 = y'_i$ и $P4 = x''_i$; $P5 = y''_i$; выигрывает первым достигший линии старта.

рости движения $d/\Delta t$. При этом в начальной зоне шириной a максимальная скорость при $P = 1$ составляет V_1 , тогда как в пустынной зоне при $y > a$ она снижается до V_2 . Побеждает, как и в настоящих автогонках, участник, выбравший наилучший путь, обеспечивший достижение финиша за минимальное время.

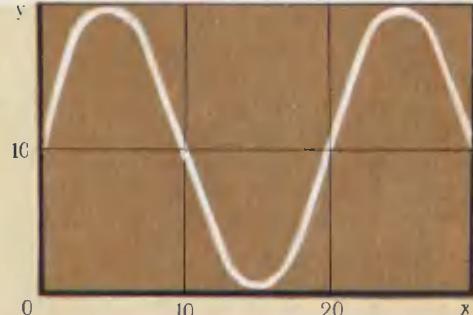


Рис. 30.

Программа 39. Игра «автопробег»

ПЗ 43	КИП4 Г4	ИП8 68	ИПА 6—	— 11	$x < 0$ 5С	10 10	ИП5 65	БП . 51	11 11
И 16 66	ИП9 69	×	12	П2 42	ИП3 63	\sin 1С	×	12	ИП8 68
И 2 62	ИП3 63	cos	1Г	×	12	ИП7 67	+	10	П7 47

Инструкция. Установить переключатель Р — Г в положение Г, перед началом игры ввести V_1 (км/ч) / 60 = P_5 ; V_2 (км/ч) / 60 = $= P_6$; a (км) = P_A ; каждый участник пробега, очистив регистр 4 и введя координаты старта $x_0 = P_7$; $y_0 = P_8$ и $P_{\max} = 1 = P_9$ проходит весь путь до финиша, нажимая перед каждым пуском программы клавиши С/П (для первого

пуска — клавиши В/О и С/П) с предварительным изменением при необходимости $P \leq 1$ в регистре 9 и вводом $\varphi = P_X$; после каждого выполнения программы $P_X = P_7 = x_i$ (км), $P_Y = P_8 = y_i$ (км), $P_4 = i$ (мин); выигрывает прошедший из старта точно в место финиша за минимальное время.

При программировании судейства таких гонок (штрафы за нарушение правил в простейшем случае можно поручить участникам) должно быть предусмотрено вычисление координат автомобилей через каждый отрезок времени Δt с учетом различных предельных скоростей при $y > a$ и $y \leq a$. Этим требованиям отвечает программа 39.

Пусть, например, $V_1 = 90$ км/ч; $V_2 = 30$ км/ч; $a = 10$ км; тогда при выборе первым участником $P = 1 = P_9$; $\varphi_1 = 45 = P_X$ по программе вычисляются координаты положения автомобиля $x_1 = y_1 = 1,06$ км через время $\Delta t = 1$ мин. Продолжая ввод только значений φ_i при максимальной тяге $P = 1$ и огибая озера, участники приближаются к финишу, для точного попадания в который им следует уменьшить P .

Сложнее и интереснее моделирование управления парусной яхтой. При постоянной силе ветра со скоростью V_B в направлении отрицательных значений координаты y (рис. 32) давление ветра на парус яхты $P = kV_B \cos(\varphi + \psi)$, где k — коэффициент, зависящий от формы паруса и плотности воздуха; φ — курсовой угол (угол направления движения яхты, определяемый поло-

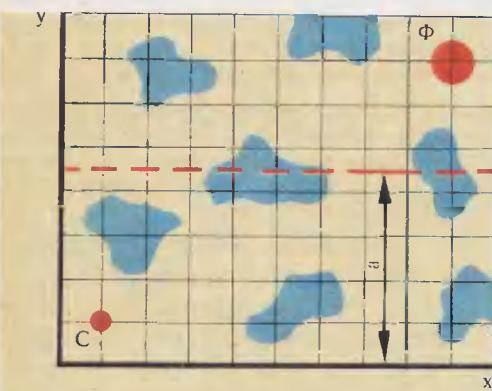


Рис. 31.

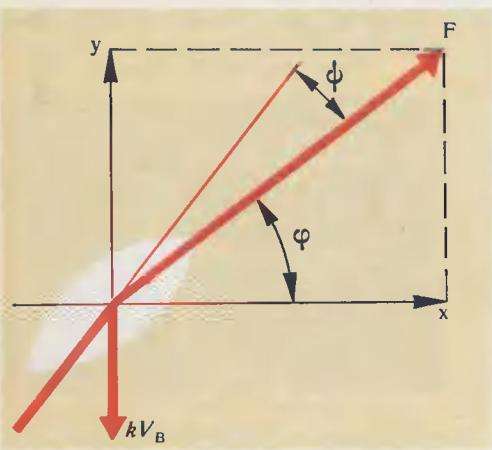


Рис. 32.

Программа 40. Игра «парусная регата»

P6 46	XY 14	P5 45	+	10	cos 1Г	ИП6 66	sin 1С	×	12	ИП9 69	×	12
П3 43	КИП4 Г4	ИП3 63	ИП5 65	sin 1С	×	12	ИП8 68	+	10	П8 48	ИП3 63	
ИП5 65	cos 1Г	×	12	ИП7 67	+	10	П7 47	С/П 50	БП 51	00 00		

Инструция. Перед началом игры установить переключатель Р — Г в положение Г и ввести коэффициент kqV_b в регистр 9; участники состязания по очереди проводят яхту по установленной трассе, предварительно очистив регистр 4 (выполняющий функции часов для отсчета суммарного времени $i\Delta t$ движения яхты) и занося координаты точки старта $x_0 =$

движением руля), отсчитываемый относительно положительного направления оси x ; ψ — угол установки паруса, отсчитываемый против часовой стрелки относительно направления движения яхты.

В пренебрежении инерционностью яхты и сопротивлением воды перемещение яхты определяется силой тяги $F = P \sin \psi = kV_b \cos(\varphi + \psi) \sin \psi$, зависящей от положения руля и паруса. За время Δt яхта проходит с некоторой средней скоростью расстояние с проекциями на координатные оси $x = qF \cos \varphi$ и $y = qF \sin \varphi$, где q — коэффициент пропорциональности между силой тяги и скоростью движения. При выборе $\Delta t = 1$ координаты яхты через каждый отрезок времени определяются по формулам $x_i = x_{i-1} + qkV_b \cos(\varphi_i + \psi_i) \sin \psi_i \cos \varphi_i$; $y_i = y_{i-1} + qkV_b \cos(\varphi_i + \psi_i) \sin \psi_i \sin \varphi_i$.

Так как яхтсмену приходится непрерывно управлять положением руля и паруса, то в реализующей вычисления по этим формулам программе 40 предусмотрен ввод значений φ_i и ψ_i перед каждым выполнением программы.

Величину kqV_b целесообразно выбирать в соответствии с реальными характеристиками движения яхт. Так

$= P7$, $y_0 = P8$; перед каждым выполнением программы ввести $\varphi_i = PY$, $\psi_i = PX$ и нажать клавиши В/О и С/П (для первого пуска обязательно) или С/П; после каждого i -го выполнения программы $PX = P7 = x_i$, $PY = P8 = y_i$, $P4 = i$; победитель определяется по минимальным затратам времени $i\Delta t$ на прохождение всей трассы.

как эта величина равна максимальному расстоянию, проходимому яхтой за время $\Delta t = 1$ мин при $\varphi = -90^\circ$ и $\psi = 0$, то приняв максимальную скорость яхты $V_{\max} = 5$ м/с, получим значение $kqV_b = V_{\max} \Delta t = 300$ м.

Яхта может двигаться (лавировать) и против ветра при изменениях положения руля и паруса. Приняв, например, $300 = P9$; $0 = P4 = P7 = P8$, при $\varphi_1 = 60^\circ$ и $\psi_1 = 15^\circ$ получим $x_1 = 10$ м; $y_1 = 17,4$ м; при $\varphi_2 = 120^\circ$ и $\psi_2 = -15^\circ$ получим $x_2 = 0$; $y_2 = -34,8$ м и т. д. Приобретя достаточные навыки в управлении яхтой, можно приступить к соревнованиям. Для этого следует начертить карту акватории с координатной сеткой и очертаниями берега и островов и выбрать трассу для регаты (гонок под парусом). Участники регаты по очереди проводят яхту к месту финиша, и победитель определяется по минимальным затратам времени ($P4 = i$).

Можно усложнить программу 40, организовав автоматическое занесение в память координат яхт, управляемых каждым из участников. Однако это не всегда целесообразно, так как ограничивает число участников и увеличивает время выполнения программы.

Программа 41. Игра «авиационный перелет»

П2 42	ХY 14	П1 41	sin 1C	ИП6 66	\times^* 12	ИП7 67	+	П7 47	КИП4 Г4
ИП1 61	cos 1Г	ИП6 66	\times 12	П3 43	ИП2 62	sin 1C	\times 12	ИП9 69	+
П9 49	ИП3 63	ИП2 62	cos 1Г	\times 12	ИП8 68	+	П8 48	С/П 50	БП 51
00 00	\uparrow 0Е	Cx 0Г	П4 44	П7 47	П8 48	П9 49	\rightarrow 25	БП 51	00 00

Инструкция. Установить переключатель Р — Г в положение Г; для взлета «пилот» вводит $d = 2 = P_6$ (что соответствует скорости 120 км/ч), угол (между направлением взлетной полосы и осью x) $\varphi_0 = PX$, допустимый угол подъема $\theta_0 = PY$ и нажимает клавиши БП 3 1 С/П; в последующем перед каждым выполнением программы «пилот» вводит $d_i = P_6$;

$\varphi_i = PX$; $\theta_i = PY$ (в пределах, допускаемых выбранными правилами игры) и нажимает клавиши В/О и С/П или С/П; после каждого i -го выполнения программы $PX = P_8 = x_i$; $PY = P_9 = y_i$; $P7 = h_i$ (высота полета); $P4 = i\Delta t$ (время полета); при выборе d следует учитывать, что при $\Delta t = 1$ мин значение $d = 10$ км соответствует скорости 600 км/ч.

Программа 42. Игра «навесная стрельба»

КИП4 Г4	ХY 14	2 02	\times 12	sin 1C	ИП8 68	\times 12	ИП7 67	$-$ 11	П5 45
x^2 22	\sqrt 21	ИП9 69	$-$ 11	$x < 0$ 1C	17 17	\sqrt 21	ИП5 65	С/П 50	БП 51
00 00									

Инструкция. Установить переключатель Р — Г в положение Г, очистить регистр 4, ввести дальность цели $d_u = P_7$, максимальную дальность стрельбы $d_{max} = P_8$, радиус поражения $\Delta = P_9$; перед каждым выполнением программы вводить угол возвышения орудия $\theta = PX$ и нажимать клавиши В/О и С/П (для

первого пуска программы обязательно) или С/П; после каждого выполнения программы высвечивается расстояние от цели до места падения снаряда (при недолете с отрицательным знаком) или, при поражении цели, символ ЕГГОГ; число затраченных снарядов (выполнений программы) хранится в регистре 4.

соответствовать направлению взлетной полосы.

Моделирование полетов самолета с постоянной скоростью на каждом отрезке времени Δt обеспечивается программой 41.

Перед «полетами» по программе 41 следует составить авиационную карту с координатной сеткой (точка старта соответствует началу координат с $x_0 = y_0 = 0$), направлениями и размерами посадочных полос. Условия игры можно разнообразить, дополнив их, например, требованием совершение

круг перед посадкой или придерживаться при перелетах заданных воздушных коридоров и изменяя ограничения на выбор d , φ и ψ .

Наиболее ярко выражены конфликтные ситуации, являющиеся существом игр, в военных операциях, и поэтому различные игровые автоматы имитируют боевые действия чаще, чем спортивные состязания. Одним из простейших боевых действий является артиллерийская стрельба, при которой снаряд, вылетевший из орудия с начальной скоростью V под углом θ относительно поверхности Земли, пролетает по баллистической кривой и падает (без учета сопротивления воздуха) на расстоянии $d = V^2 \sin 2\theta / g$, где $g = 9,81$ м/с² (рис. 33).

Для моделирования артиллерийской стрельбы эту формулу удобно записать в виде $d = d_{\max} \sin 2\theta$, где $d_{\max} = V^2/g$ — максимальная дальность стрельбы, соответствующая углу возвышения $\theta = 45^\circ$.

Выбрав d_{\max} и радиус поражения Δ цели при разрыве снаряда, можно моделировать артиллерийскую стрельбу с помощью программы 42, пригодной для различных игр.

Программу 42 можно использовать в различных вариантах игры. При «учебных стрельбах» участники игры по очереди «ведут огонь» до поражения цели и выигрывает «артиллерист», затративший минимальное число снарядов. При «артиллерийской дуэли» участники игры по очереди «ведут огонь», и выигрывает первым поразивший противника.

Игру можно усложнить, предполагая, например, движение цели в направлении орудия с постоянной скоростью. Для этого достаточно организовать в программе уменьшение дальности до цели после каждого ее выполнения. Такую программу, подобную программе 43, можно использовать как для отражения «танковой атаки» за минимальное число «выстрелов», так и для моделирования боя между орудием и танком, по очереди обменивающихся выстрелами.

Подобные игры можно усложнить, учитывая не только дальность до цели, но и азимут цели φ (угол между направлением на цель и начальным направлением). Кроме того, можно усложнить и задачу артиллеристов, сообщая им лишь расстояние от цели до места падения снаряда

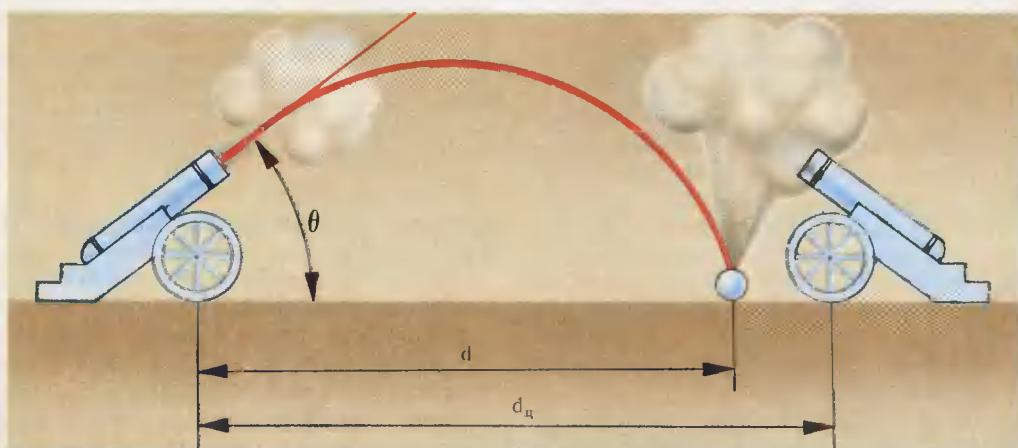


Рис. 33.

Программа 43. Игра «танковая атака»

КИП4 Г4	ХY 14	2 02	\times 12	sin 1C	ИП9 69	\times 12	ИП7 67	ИП6 66	— 11
ИП7 47	— 11	ИП5 45	x^2 22	$\sqrt{}$ 21	ИП8 68	— 11	$x < 0$ 5C	20 20	$\sqrt{}$ 21
ИП5 45	С/П 50	БП 51	00 00						

Инструкция. Установить переключатель Р — Г в положение Г; ввести начальную дальность до цели $d_{u0} = P7$, радиус поражения цели $\Delta = P8$, максимальную дальность стрельбы $d_{max} = P9$, изменение расстояния до цели за время полета снаряда $\Delta d_u = P6$; очистить регистр 4; перед каждым пуском программы

нажатием клавиши С/П (первый раз — клавиши В/О и С/П) вводить угол возвышения $\theta = RX$; после выполнения программы высвечивается расстояние между целью и местом падения снаряда или, при поражении цели — сигнал ЕГТОГ; число выстрелов хранится в регистре 4.

Программа 44. Игра «артиллерийский бой»

П9 49	ХY 14	2 02	\times 12	sin 1C	ИП7 67	\times 12	П8 48	КИП4 Г4	ИП8 68
x^2 22	ИП5 65	x^2 22	+	10	ИП5 65	ИП8 68	\times 12	2 02	\times 12
ИП6 66	— 11	cos 1Г	\times 12	— 11	$\sqrt{}$ 21	П2 42	ИП1 61	— 11	$x < 0$ 5C
32 32	$\sqrt{}$ 21	ИП2 62	С/П 50	БП 51	00 00	П4 44	+	10	П5 45
\times 12	П6 46	ИП5 65	ИП4 64	1 01	+	10	\div 13	5 05	+
Сх 0Г	П4 44	С/П 50	БП 51	00 00					П5 45

Инструкция. Установить переключатель Р — Г в положение Г; ввести максимальную дальность стрельбы $d_{max} = P7$, радиус поражения $\Delta = P1$, часы начала игры $H = PY$, минуты начала игры $M = RX$ и нажать клавиши БП 3 6 С/П, что приведет к высвечиванию нуля; перед каждым пуском про-

граммы вводить угол возвышения $\theta = PY$ и азимут $\varphi = RX$ и нажимать клавишу С/П; после выполнения программы высвечивается расстояние от места падения снаряда до цели, или при поражении цели, сигнал ЕГТОГ; число выполнений программы хранится в регистре 4.

$$r = \sqrt{d^2 + d_u^2 - 2dd_u \cos(\varphi - \varphi_u)},$$

где d_u — расстояние от орудия до цели; d — расстояние от орудия до места падения снаряда; φ — азимут, выбранный артиллеристами; φ_u — истинный азимут цели.

Для усложнения задачи артиллеристов в программе 44, реализующей вычисления по этой формуле, даль-

ность и азимут цели выбирают по формулам $d_u = (H + M)/(M + 1)$; $\varphi_u = 2(H + M)$, где H и M — часы и минуты начала игры.

Подобные программы пригодны как для «учебных стрельб», так и для «артиллерийских дуэлей», но их можно еще более усложнить и приблизить моделирование к реальным условиям, учитывая движение цели

Программа 45. Игра «морское сражение»

П1 41	XY 14	П2 42	2 02	\times	12	sin 1C	ИП5 65	\times	12	П3 43	КИП4 Г4
ИП5 65	$\sqrt{ }$ 21	2 02	0 00	\times	12	ИП2 62	sin 1C	\times	12	ИП9 69	\times 12
ИП7 67	ИП8 68	cos 1Г	\times	12	XY 14	—	ПА 4—	x^2	22	ИП7 67	ИП8 68
sin 1C	\times	12	x^2	22	+	10	$\sqrt{ }$	21	П7 47	ИПА 6—	\div
П8 48	ИП7 67	x^2	22	ИП3 63	x^2	22	+	10	ИП8 68	ИП1 61	—
ИП7 67	\times	12	ИП3 63	\times	12	2 02	\times	12	—	$\sqrt{ }$	ПВ 4L
— 11	$x < 0$ 5C	68	68	5 05	0 00	5 05	БП 51	69	69	ИПВ 6L	С/П 50
БП 51	00 00	\uparrow	0E	Cx 0Г	П4 44	\rightarrow	БП 51	00 00			

Инструкция. Установить переключатель Р — Г в положение Г; ввести d_{\max} (км) = = Р5; Δ (км) = Р6; d_{k0} (км) = Р7; $\alpha(\dots)$ = Р8; V_k (км/с) = Р9; перед каждым пуском программы вводить $\theta(\dots)$ = РY; $\varphi(\dots)$ = RX и нажимать клавиши В/О и С/П или С/П (для первого пуска — клавиши БП 7 2 С/П); после выполне-

ния программы высвечивается расстояние от корабля до места падения снаряда или, при поражении, сигнал SOS (505); число выстрелов хранится в регистре 4, расстояние до места падения снаряда — в регистре 3, истинные значения d_k и φ_k — соответственно в регистрах 7 и 8.

под углом относительно направления на орудие. Так, при движении корабля со скоростью V_k за время полета снаряда, выпущенного из орудия при угле возвышения θ , корабль пройдет по своему курсу расстояние $a = V_k t$, а время полета снаряда составит $t = 2V \sin \theta / g = 2 \sin \theta \sqrt{d_{\max} / g} \approx \approx 20 \sin \theta \sqrt{d_{\max}}$. Если обозначить $b = d \sin \alpha$; $c = d \cos \alpha$ (где α — курсовой угол корабля в соответствии со схемой, показанной на рис. 34) и отсчитывать азимут φ относительно курса корабля, то ко времени падения снаряда дальность до корабля составит $d_k = \sqrt{b^2 + (c - a)^2}$, а азимут корабля $\psi_k = \arccos((c - a) / d_k)$ при расстоянии до места падения снаряда r , определяемого как и для предыдущей программы.

Эти соотношения использованы для вычислений, реализуемых по программе 45, где при измерении углов в градусах, а расстояний в километрах, для определения времени в секундах скорость корабля всегда должна задаваться в километрах в секунду.

Пусть исходные данные к программе 45 выбраны следующими: $d_{\max} = 30$ км; $\Delta = 0,1$ км; $d_{k0} = 15$ км; $\alpha = 45^\circ$; $V_k = 21,6$ км/ч = $6 \cdot 10^{-3}$ км/с. Приняв $\theta = 20^\circ$, $\varphi = 48^\circ$ и нажав клавиши БП 7 2 С/П, после выполнения программы (время счета около 30 с) получим $PX = 4,49$. При $\theta = 15^\circ$, $\varphi = 50^\circ$ получим $PX = 1,05$ и, продолжив «стрельбу», поразим корабль, если он не успеет уйти за пределы досягаемости орудия с выбранной максимальной дальностью стрельбы.

Условия военных игр можно разнообразить до бесконечности, моделируя действия различных боевых средств и тактику их применения. Поэтому рассмотрим лишь моделирование торпедных атак, связанных с относительно сложными расчетами.

Пусть корабль, обнаруженный подводной лодкой на расстоянии d_0 , движется со скоростью V_k под углом α относительно направления на лодку (рис. 35). Командир подлодки должен выпустить торпеду с упреждением так, чтобы при скорости движения V_t ее курс пересекся в определенный момент времени с курсом корабля. Торпеда, выпущенная под углом φ , пересекает курс корабля, пройдя расстояние $b = d_0 \sin \alpha / \sin(\alpha + \varphi)$ за время $t = b/V_t$. За это же время корабль пройдет по своему курсу расстояние $a_k = V_k t = b V_k / V_t = V_k d_0 \times \sin \alpha / V_t \sin(\alpha + \varphi)$ при расстоянии до места пересечения курсов $a = d_0 \times \sin \varphi / \sin(\alpha + \varphi)$ и до места пересечения курса торпедой $\Delta a = a - a_k$. При последующих торпедных залпах $\Delta a_n = d \sin \varphi_n / (\sin(\alpha + \varphi_n)) - (V_k d \sin \alpha / V_t) \sum_{i=4}^n \sin(\alpha + \varphi_i)^{-1}$.

Если скорость корабля больше скорости торпеды, то при больших углах встречи α торпеда может не догнать корабль. Кроме того, командир подлодки должен учитывать, что при больших отношениях V_k/V_t корабль быстро проходит зону досягаемости торпеды даже при малых углах α . Эти соображения должен учитывать «подводник», использующий программу 46.

Пусть для программы 46 выбраны значения $d_0 = 10$ км; $\alpha = 30^\circ$ и $V_k/V_t = 0,45$; $\Delta = 0,1$ км. Для первого торпедного залпа выполним $15 = = PX$ БП З 6 С/П $PX = 0,48$, для второго залпа — $25 = PX$ С/П $PX = -0,77$, для третьего — $55 = PX$ С/П $PX = 505$ (цель поражена).

В условиях многих игр, моделирующих с различной точностью реальные конфликтные ситуации, учтено влияние случайных событий. В простейших случаях моделирование случайного события бросанием монеты или кубика используют для выбора очередного хода, но в некоторых играх учитывают влияние и более реальных случайных событий. В качестве примера составим игру «арктический рейс», имитирующую проводку судна

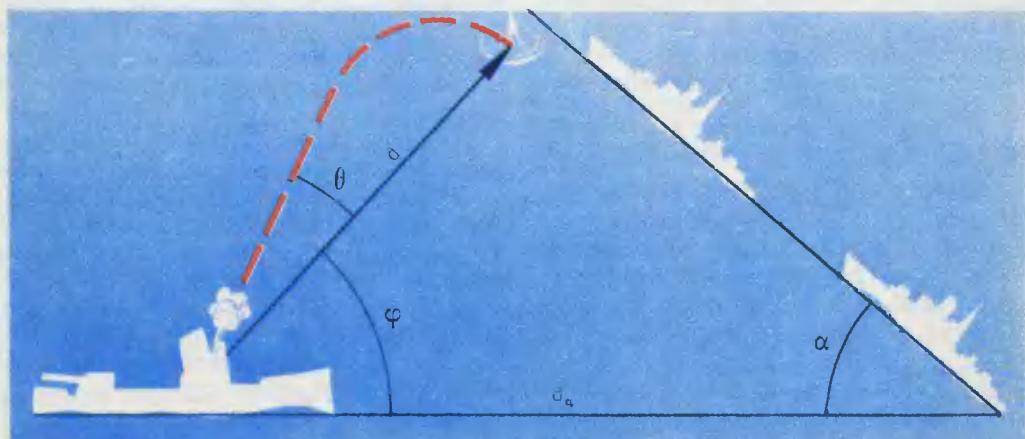


Рис. 34.

Программа 46. Игра «торпедная атака»

П5 45	КИП4 Г4	ИП5 65	ИП8 68	+	10	sin 1C	1/x 23	П1 41	ИП5 65	sin 1C
Х 12	ИП1 61	ИП2 62	+	10	П2 42	ИП3 63	Х 12	— 11	ИП7 67	Х 12
П0 40	x^2 22	$\sqrt{ } 21$	ИП6 66	— 11	$x < 0$ 5C	32	32	5 05	0 00	5 05
БП 51	33 33	ИП0 60	С/П 50	БП 51	00 00	П5 45	ИП9 69	ИП8 68	sin 1C	
Х 12	П3 43	Cx 0Г	П2 42	П4 44	B/O 52					

Инструкция. Установить переключатель Р — Г в положение Г; ввести исходные данные $d_0 = P7$; $\alpha = P8$; $V_k/V_t = P9$; половину длины корабля $\Delta = P6$; для торпедного залпа ввести $\varphi = RX$ и нажать для первого пуска программы клавиши БП 3 6 С/П, для последующих — В/О и С/П или С/П; после выполне-

ния одного северного порта в другой при условии возможного скопления льдов в прибрежных районах.

Составив карту маршрута (рис. 36), условимся что вследствие сложной ледовой обстановки в районе с координатами $x > a$, $y < b$ на каждом отрезке времени Δt равновероятно наличие тяжелого льда, препятствующего движению корабля, и условий, обеспе-

чивающих нормальное движение. Выбрав, например, маршрут 1, капитан беспрепятственно достигнет порта назначения, но если выбран маршрут 2, льды могут задержать корабль, и выбравший этот более короткий маршрут капитан рискует задержаться в пути.

Для составления программы 47, обеспечивающей штурмана корабля

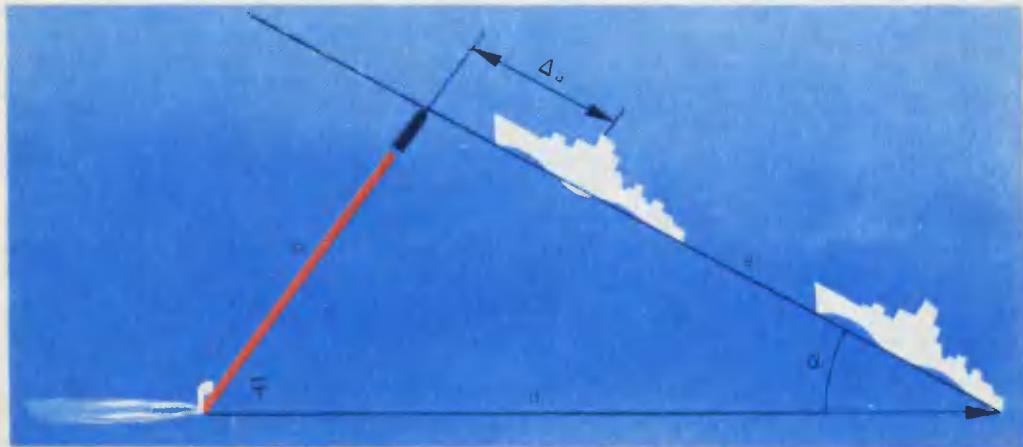


Рис. 35.

Программа 47. Игра «арктический рейс»

П7 47	\cos 1Г	ИПД 6Г	\times 12	П5 65	КИП4 Г4	ИП7 67	\sin 1С	ИПД 6Г	\times 12
П6 46	ИП9 69	ИПВ 6L	— 11	$x < 0$ 5C	39 39	ИП8 68	ИПА 6—	— 11	$x \geq 0$ 59
39 39	ИПС 6C	π 20	$+$ 10	e^x 16	ПС 4C	КИПС ГС	\rightarrow 25	ИПС 6C	— 11
ПС 4C	2 02	$1/x$ 23	— 11	$x < 0$ 5C	39 39	Сх 0Г	П5 45	П6 46	ИП6 66
ИП9 69	+	10	П9 49	ИП5 65	ИП8 68	+	П8 48	С/П 50	БП 51
									00 00

Инструкция. Установить переключатель Р — Г в положение Г; ввести $0 = P4 = PC$; выбранные значения $a = PA$; $b = PB$; $d = PD$; начальные координаты корабля $x_0 = P8$; $y_0 = P9$; перед каждым выполнением программы

вводить курсовой угол $\varphi = RX$ и нажимать клавишу С/П (первый раз — клавиши В/О и С/П); результаты выполнения программы $PX = P8 = x_i$; $PY = P9 = y_i$; $P4 = i$.

Программа 48. Оценка ответов по экзаменационному билету

КИПС ГС	— 11	$x = 0$ 5E	08 08	ИПО 60	1 01	+	10	П0 40	ИПД 6Г	ИПС 6C
— 11	$x \neq 0$ 57	20 20	ИПС 6C	1 01	+	10	ПС 4C	С/П 50	БП 51	00 00
ИПО 60	ИПС 6C	\div 13	4 04	\times 12	1 01	+	10	ИПО 60	ХУ 14	С/П 50
Сх 0Г	П0 40	1 01	ПС 4C	С/П 50	БП 51	00	00			

Инструкция. (Ввести общее число вопросов $n = PD$ ($n \leq 11$)); числа, соответствующие правильным ответам на вопросы, ввести в регистры памяти с номерами 1, 2, 3, ..., равными порядковым номерам вопросов), нажать клавиши БП 3 0 С/П, что приводит к вычислению номера 1 первого вопроса; экзаменую-

щийся вводит свой ответ в численном виде в регистр X и нажимает клавиши В/О и С/П, после чего высвечивается номер следующего вопроса; после ввода ответа на последний вопрос и выполнения программы высвечивается оценка в пятибалльной системе, а число правильных ответов хранится в регистре Y.

сведениями о координатах судна через каждый отрезок времени в зависимости от ледовой обстановки и принятых капитаном решений о курсе корабля, следует обеспечить вычисление координат по формулам

$$x_i = x_{i-1} + kd \cos \varphi_i; \quad y_i = y_{i-1} + kd \sin \varphi_i,$$

где d — расстояние, пройденное кораблем за время Δt (для простоты будем принимать его постоянным); k — коэффициент, равный 1 в районе

чистой воды и 0 в районе тяжелых льдов с вероятностью $p(1) = 0,5$.

Микро-ЭВМ можно поручить не только судейство спортивных игр, но и суждение об уровне знаний состязающихся в определенной области. Простейший способ машинной проверки знаний связан с использованием экзаменационного билета с рядом вопросов, на каждый из которых требуется ответить числом, которым может быть как число, которое

должен знать экзаменующийся, так и код (номер) правильного ответа, перечень которых дается. В этом случае достаточно составить для микро-ЭВМ программу-экзаменатор, подобную программе 48 для микрокалькулятора, сравнивающую ответы экзаменующегося с правильными ответами, хранящимися в памяти, и оценивающую знания по числу правильных ответов.

Пример экзаменационного билета для программы 48 приведен в табл. 8 со следующей последовательностью заносимых в регистры памяти 1, 2, ..., A правильных ответов: 1852; 3; 1067; 1; $e^1 = 2,718\ 281\ 8$; 4; $\lg 10 / \lg 2 = 3,321\ 928\ 1$; 1; 2,54; 2.

Эта программа достаточно строго оценивает ответы, вычисляя оценку по формуле $z = E[4m/n + 1]$, где n

и m — числа всех и правильных ответов, а E — символ целой части содержащего скобок. Пятерка высвечивается только при правильных ответах на все вопросы, а для получения тройки нужно ответить правильно не менее, чем на половину вопросов.

Квалификацию программы-экзаменатора можно повысить, дополнив ее фрагментами для оценки приближенных ответов, но возможности программируемых микрокалькуляторов ограничены вводом и выводом информации только в цифровом виде. Значительно большие возможности автоматизации не только ответов на вопросы экзаменационных билетов (которые могут сопровождаться, например, высвечиванием графиков), но и элементов обучения связаны с при-

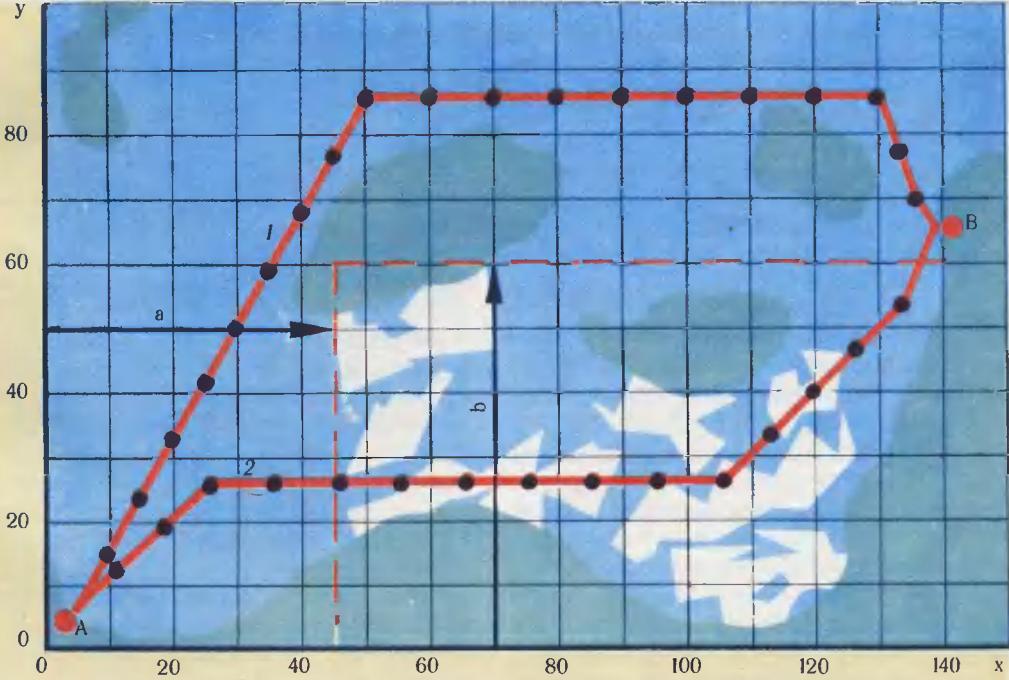


Рис. 36.

8. Экзаменационный билет №13 по общей эрудиции

№	Вопрос
1	Сколько метров в морской миle?
2	Что такое Лимпопо? Ответы: 1. Озеро на Ямайке. 2. Остров в Полинезии. 3. Река в Африке. 4. Таитянское божество. 5. Персонаж индейского мифа.
3	Сколько метров в версте?
4	Сколько рек вытекает из Байкала?
5	Чему равно основание натуральных логарифмов с точностью до восьми значащих цифр?
6	Автор (литературный герой) известных баллад о Гавриле? Ответы: 1. Полосов. 2. Кизлярский. 3. Изнуренков. 4. Ляпис. 5. Остап Бендер.
7	Чему равен $\log_2 10$ с точностью до восьми значащих цифр?
8	Что такое гарнец? Ответы: 1. Древняя мера сыпучих тел. 2. Вид гончарных изделий. 3. Житель местности в Германии. 4. Металлургический термин.
9	Сколько сантиметров в дюйме с точностью до трех знаков?
10	Какой фунт равен 0,456 кг? Ответы: 1. Русский. 2. Английский. 3. Лапуасский.

менением персональных микро-ЭВМ. Например, при использовании такой машины с языком программирования БЕЙСИК начало программы для экзамена по билету № 13 (табл. 8) с «под-

сказками» может быть составлено в следующем виде:

NEW

2 PRINT' ВОПРОС 1. СКОЛЬКО МЕТРОВ В МОРСКОЙ МИЛЕ.
ДЛЯ ОТВЕТА НАБЕРИТЕ НА КЛАВИАТУРЕ 6 LET X =
RUN
С УКАЗАНИЕМ СПРАВА ОТ ЗНАКА РАВЕНСТВА ТРЕБУЕМОГО ЧИСЛА'

4 STOP

8 IF X = 1600 THEN 20
10 IF X = 1852 THEN 22
12 LET K = ABS(X - 1852)
14 IF K < 10 THEN 26
16 PRINT' ОТВЕТ ОШИБОЧЕН. ДЛИНА МОРСКОЙ МИЛИ 1852 М. ДЛЯ ВЫЗОВА ВТОРОГО ВОПРОСА НАБЕРИТЕ НА КЛАВИАТУРЕ
30 LET N = 2
RUN

18 STOP

20 PRINT' ВЫ УКАЗАЛИ ДЛИНУ СУХОПУТНОЙ МИЛИ. ПОВТОРИТЕ НАВОРОТВЕТА НА ПЕРВЫЙ ВОПРОС'

21 STOP

22 PRINT' ОТВЕТ ТОЧЕН. ДЛЯ ВЫЗОВА ВТОРОГО ВОПРОСА НАБЕРИТЕ
30 LET N = 2

24 STOP RUN'

26 PRINT' ОТВЕТ НЕТОЧЕН. ДЛИНА МОРСКОЙ МИЛИ 1852 М. ДЛЯ ВЫЗОВА ВТОРОГО ВОПРОСА НАБЕРИТЕ НА КЛАВИАТУРЕ
30 LET N = 2

28 STOP RUN'

Подобным образом составляются программы работы микро-ЭВМ и для обучения решению конкретных задач. Возможность вывода на дисплей графиков и изображений требуемых предметов значительно повышает эффективность подобных программ.

Глава 7

ИГРЫ С МОДЕЛИ- РОВАНИЕМ ДВИЖЕНИЯ ИНЕРЦИОННЫХ ОБЪЕКТОВ

При моделировании движения различных объектов до сих пор не учитывалось влияние их массы, так как по условиям рассмотренных задач предполагалась постоянная скорость движения в течение достаточно больших интервалов времени. Однако в большинстве практических задач это предположение неприемлемо и приходится учитывать инерционность движущихся объектов.

Согласно второму закону Ньютона с учетом силы R сопротивления среды движение тела массой m под воздействием силы тяги F описывается уравнением $ma = F - R$, где a — ускорение (скорость изменения скорости) движущегося тела.

Во время движения тела по поверхности появляется сила трения скольжения или качения, но она практически постоянна при небольших скоростях, а при больших — значительно меньше силы сопротивления среды (воздуха или воды) и может быть учтена соответствующим изменением силы тяги. Сила сопротивления воздуха или воды пропорциональна квадрату скорости и с учетом ее направления уравнение движения $ma = F - AV|V|$, где V — мгновенная скорость движущегося тела; A — коэффициент пропорциональности.

Средняя скорость движения $V = \Delta x / \Delta t$ в направлении координаты x определяется приращением Δx расстояния за время Δt и при стремлении Δt к нулю равна мгновенной скорости. При моделировании движения инерционных объектов на ЭВМ уравнение движения последовательно решают для моментов времени $t_i = t_{i-1} + \Delta t$, отличающихся на постоянную величину Δt , называемую шагом. Величина Δt не может быть равной нулю и должна выбираться как можно большей для уменьшения объема вычислений при моделировании движения в течение заданного интервала времени. Поэтому шаг Δt выбирают как максимальный отрезок времени, в течение которого ускорение практически не изменяется. Так как изменения ускорения $a = (F - P)/m$ при прочных равных условиях обратно пропорциональны массе, то при моделировании объектов с большей массой можно выбирать и большие значения шага Δt .

Выбор шага Δt связан и с допустимыми изменениями силы тяги за время Δt . Так, при малых скоростях сила сопротивления среды пренебрежимо мала, и приращение ускорения $a_i - a_{i-1} = (F_i - F_{i-1})/m$. Следовательно, выбор Δt по условию $\Delta a \leq \varepsilon$ соот-

всегда предельным изменениям силы тяги на каждом шаге $\Delta F \leq m \cdot \epsilon$.

При выборе достаточно малого шага Δt мгновенную скорость определяют по конечной разности расстояний согласно формуле $V_i = (x_i - x_{i-1})/\Delta t = \Delta x_i/\Delta t$, а ускорение в момент времени $t_{i+1} = t_i + \Delta t$ по формуле $a_{i+1} = (\Delta V_{i+1} - \Delta V_i)/\Delta t = (\Delta x_{i+1}/\Delta t - \Delta x_i/\Delta t)/\Delta t = (\Delta x_{i+1} - \Delta x_i)/(\Delta t)^2$.

Подставив эти выражения в уравнение движения, преобразуем его в разностное уравнение

$$m(x_{i+1} - x_i - \Delta x_i)/(\Delta t)^2 = F_i - A\Delta x_i|\Delta x_i|/(\Delta t)^2.$$

Это уравнение несложно представить расчетной формулой для вычисления очередного значения координаты расстояния.

$$x_{i+1} = x_i + \Delta x_i + F_i(\Delta t)^2/m - A\Delta x_i|\Delta x_i|/m.$$

Коэффициент A в этой формуле может быть определен по известной для каждого моделируемого объекта максимальной скорости V_{\max} , достигаемой при максимальной силе тяги F_{\max} . Так как эта скорость ограничена силой сопротивления среды, то условие $F_{\max} = AV_{\max}^2 = 0$ определяет и коэффициент $A = F_{\max}/V_{\max}^2$.

Полученную расчетную формулу для последовательного вычисления x_{i+1} легко реализовать в программе работы любой универсальной микроЭВМ. Для уменьшения объема вычислений целесообразно выбрать шаг Δt , равный одной из основных единиц времени (секунда, минута, час) или кратным ей в соответствии с массой моделируемого объекта. При этом удобно задавать не величины F_i и A , а их нормированные значения $P_i = F_i(\Delta t)^2/m$ и $B = A/m$. В этом случае расчетная формула упрощается:

$$x_{i+1} = x_i + \Delta x_i + P_i - B\Delta x_i|\Delta x_i|.$$

Следует учитывать, что составленная формула для моделирования

движения по прямой не учитывает действия тормозных устройств. Поэтому она непосредственно пригодна для моделирования движения объектов (например, речных или морских самоходных средств), не имеющих таких устройств и останавливаемых изменением знака силы тяги (задним ходом).

При выборе $\Delta t = 1$ с по составленной расчетной формуле можно достаточно точно моделировать движение судов с небольшой массой (катеров или моторных лодок), но для моделирования движения судов с водоизмещением в сотни и тысячи тонн допустимо принимать $\Delta t = 1$ мин. Первому из этих условий отвечает программа 49 при $\Delta t = 1$ с для моделирования движения по прямой судов малого водоизмещения. Однако при выборе объектом моделирования судна с большим водоизмещением также можно воспользоваться этой программой, приняв большее значение Δt и соответственно вычислив значения коэффициентов и размерность расстояний и скоростей.

Пусть для гонок по программе 49 выбран катер массой $m = 1$ т, максимальной силой тяги движителя (винта) $F_{\max} = 1000$ Н, допустимыми изменениями силы тяги $\Delta F/c \leq 200$ Н/с и максимальной скоростью $V_{\max} = 36$ км/ч = 20 м/с. В связи с небольшой массой катера выберем $\Delta t = 1$ с и вычислим $P_{\max} = F_{\max}/m = 1$; $\Delta P \leq 0,2$ и $B = P_{\max}/V_{\max}^2 = 0,25 \cdot 10^{-3}$.

Задавая нормированную силу тяги перед каждым пуском программы $P_i = 0,2; 0,4; 0,6; 0,8; 1; 1; 1; 1; 1; ...$, получим соответственно $x_{i+1} = 0,2; 0,8; 2; 4; 7; 10,9; 15,9; 21,7; 28,5; 36,2; ...$; $V_{i+1} = \Delta x_{i+1}/1 = 0,2; 0,6; 1,2; 3; 3,9; 4,9; 5,86; 6,78; 7,66; ...$. Таким образом, за первые 10 с моделируемый катер проходит расстояние 36,2 м, достигая при этом скорости

Программа 49. Игра «гонки по прямой на катерах»

КИП4 Г4	ИП9 69	ИП7 67	\uparrow ОЕ	x^2 22	$\sqrt{\quad}$ 21	$\times.$ 12	ИПВ 61	\times 12	$-$ 11
ИП7 67	+	10	ИП7 47	ИП7 67	ИП8 68	+	10	П8 48	С/П 50
Сх ОГ	П4 44	П7 47	П8 48	БП 51	00 00			БП 51	00 00

Инструкция. (Ввести $B = P_{\max}/V_{\max}^2 = PB$); ввести начальное значение нормированной силы тяги $P_0 = P9$ и нажать клавиши БП 2 0 С/П для старта с пункта с координатами $x_0 = y_0 = 0$; в дальнейшем, изменяя при необходимости $P_i = P9$, для каждого пуска программы нажимать клавиши В/О и С/П или

только С/П; после выполнения программы $PX = P8 = x$; $PY = P7 = \Delta x$; $P4 = i$ (для заднего хода вводить $P < 0$); выигрывает участник игры, прошедший заданное расстояние и достигший финиша при скорости причаливания $V \leq 1$ м/с за минимальное время $t = i\Delta t$.

Программа 50. Игра «школа капитанов для судов на подводных крыльях»

КИП4 Г4	ИП7 67	ИПА 6—	$-$ 11	$x \geq 0$ 59	09 09	ИПВ 61	БП 51	10 10	ИПС 6С
ИП7 67	\uparrow ОЕ	x^2 22	$\sqrt{\quad}$ 21	\times 12	\times 12	ИП7 67	ХУ 14	$-$ 11	ИП9 69
+	10	П7 47	ИП7 67	ИП8 68	+	10	П8 48	С/П 50	БП 51
П4 44	П7 47	П8 48	БП 51	00 00			00 00	Сх ОГ	

Инструкция. (Принять $V_k = PA$; $B_1 = -PB$; $B_2 = PC$); для каждого «рейса» ввести начальное значение $P_0 = P9$ и нажать клавиши БП 2 9 С/П, в дальнейшем, изменяя при необходимости $P_i = P9$, пускать программу нажатием клавиш В/О и С/П или С/П; после каждого

1-го выполнения программы $PX = P8 = x$; $PY = P7 = \Delta x$; $P4 = i$; выигрывает «капитан», прошедший заданное расстояние и причаливший с допустимой скоростью за минимальное время $t = i\Delta t$.

$V = 7,66$ м/с. Своевременно уменьшив тягу перед приближением к финишу и «отработав задним ходом», с требуемой скоростью подходим к причалу.

Сила сопротивления среды некоторых движущихся объектов может изменяться в зависимости от режима движения. Примером являются суда на подводных крыльях, корпус которых при достижении определенной скорости V_k и соответствующей подъемной силы подымается над поверхностью воды, сопротивление среды уменьшается и максимальная скорость движения увеличивается. Эта

особенность судов на подводных крыльях моделируется программой 50, при выполнении которой при малых и больших скоростях принимаются различные значения коэффициента силы сопротивления воды (B_1 при $V < V_k$ и B_2 при $V > V_k$), что обеспечивает возможность достижения скорости $V_{1\max}$ при опущенном корпусе и значительно большей скорости $V_{2\max}$ при поднятом корпусе.

Предположим, что моделируется движение теплохода на подводных крыльях массой $m = 100$ т, максимальной силой тяги $F_{\max} = 50\,000$ Н, допустимыми изменениями силы тяги

Программа 51. Игра «автогонки по прямой»

ИПД 6Г	$x = 0$ 5Е	13 13	ИП7 67	\uparrow 0Е	\uparrow 0Е	x^2 22	$\sqrt{ }$ 21	\times 12	ИПВ 6L
\times 12	— 11	П7 47	КИП4 Г4	ИП7 67	ИПД 6Г	— 11	$x < 0$ 5С	20 20	Cx 0Г
ИП9 69	+	10 47	ИП8 68	+	10	П8 48	ИП7 67	3 03	, 0—
\times 12	C/П 50	БП 51	00 00	Cx 0Г	П4 44	П7 47	П8 48	ПД 4Г	B/O 52
Cx 0Г	5 05	ПД 4Г	БП 51	00 00					

Инструкция. Ввести $B = F_{\max}(\Delta t)^2 / m V_{\max}^2 = PB$; перед каждым заездом «отпустить тормоз», очистив регистр Д и «часы» на регистре 4; ввести координаты старта $x_0 = -P7'$, $y_0 = P8$; ввести $P_0 = P9$ и нажать клавиши В/О и С/П (или, при автоматической очистке регистров 4, 7, 8 и Д, клавиши БП 3

$\Delta t \leq 1000$ Н/с, скоростью, при которой корпус подымается над водой, $V_k = 25$ км/ч, максимальными скоростями движения $V_{1\max} = 25$ км/ч при опущенном и $V_{2\max} = 72$ км/ч при поднятом корпусе.

Для уменьшения числа выполнений программ примем $\Delta t = 1$ с. Тогда $V_k = 6,9444$ м/с; $V_{1\max} = 8,33333$ м/с; $V_{2\max} = 20$ м/с; нормированная сила тяги $P_{\max} = F_{\max}(\Delta t)^2 / m = 50$; $\Delta P \leq 10$; $B_1 = P_{\max} / V_{1\max}^2 = 7,2 \cdot 10^{-3}$; $B_2 = P_{\max} / V_{2\max}^2 = 1,25 \cdot 10^{-3}$.

При этих исходных данных для значений $P_i = 10; 20; 30; 40; 50; 50; 50; 50; 50; 50; 50; 50; 50; 50; 50; ...$ по программе 50 получим $x_{i+1} = 10; 39,28; 92,39; 165,2; 281,4; 430,7; 602,11; 786,81; 978,87; 1174,8; 1372,8; 1571,7; ...$; $\Delta x_{i+1} = 10; 29,28; 53,11; 72,8; 116,2; 149,3; 179,44; 184,7; 192,06; 195,9; 197,95; 198,97; ...$. Следовательно, моделируемый теплодвигатель примерно через 35 с выходит на подводные крылья и через 2 мин достигает максимальной скорости, пройдя свыше 1570 м.

Составленная расчетная формула

4 С/П); далее, изменения или сохраняя $P_i = P9$, нажимать для пуска программы клавиши В/О и С/П или С/П; для торможения нажать клавиши БП 4 0 С/П и далее С/П, уменьшая силу тяги до нуля; после каждого выполнения программы (при $\Delta t = 1$ с) $RX = V_i$; $P7 = \Delta x_i$; $P8 = x_i$; $P4 = i$.

пригодна и для моделирования сухопутных самодвижущихся средств, но в этом случае необходимо предусмотреть моделирование действия тормозных устройств. Торможение автомобиля при выборе $\Delta t = 1$ с с учетом зависимости тормозного пути от скорости можно моделировать следующим простым алгоритмом:

- Если $V < 5$ м/с, то перейти к шагу 2, иначе к шагу 3.
- Принять $V = P$ до выключения тормоза.
- Принять $V = V - 5 + P$ и перейти к шагу 1.

Силу торможения для этого алгоритма можно уменьшить, заменив 5 на 4 или 4,5. Этот алгоритм, предусматривающий и возможность «езды на тормозе», реализован в программе 51, предусматривающей вывод на индикатор значений скорости с привычной для водителей размерностью километры в час.

Выберем для гонок спортивный автомобиль массой $m = 800$ кг; максимальной силой тяги $F_{\max} = 3200$ Н; допустимыми изменениями силы тяги

$\Delta F \leq 400$ Н/с и максимальной скоростью $V_{\max} = 288$ км/ч = 80 м/с. Для $\Delta t = 1$ с находим $P_{\max} = 4$; $\Delta P \leq 1$; $B = P_{\max}/V_{\max}^2 = 6,25 \cdot 10^{-4}$. По программе 51 для первых 10 с движения при $P_i = 1, 2, 3, 4, 4, 4, 4, 4, 4, 4, \dots$ получим соответственно $V_{i+1} = 3,6; 10,8; 21,6; 35,9; 50,1; 64; 77,7; 91,1; 104; 116,6; \dots$ км/ч, $x_{i+1}(\text{м}) = 1; 4; 10; 20; 39,9; 51,7; 73,3; 98,55; 127,5; 159,8; \dots$ Следовательно, за 10 с автомобиль с выбранными характеристиками разгоняется до скорости 116,6 км/ч, проходя расстояние 160 м.

Для проверки тормозов нажмем клавиши БП 4 0 С/П и затем С/П, уменьшая перед каждым пуском силу тяги до нуля; $P_i = 3, 2, 1, 0, 0, 0, 0, \dots$, получая соответственно $V_{i+1} = 109,4; 98,6; 84,2; 66,2; 48,15; 30,2; 12,2; 0; \dots$ км/ч; $x_{i+1} = 190,2; 217,6; 240,9; 259,3; 271,3; 281,1; 284,5; 284,5; \dots$ Таким образом, при выбранной скорости автомобиль полностью тормозится за 8 с при 125 м тормозного пути, что достаточно хорошо соглашается со свойствами настоящего автомобиля. Для продолжения движения необходимо «отпустить тормоз», очистив регистр Д.

Если движение по прямой морских и сухопутных самодвижущихся средств отличается лишь способом остановки, то при движении с изменением направления силы тяги эти отличия становятся более существенными. При выбранных допущениях ($\Delta x_i \approx \Delta x_{i+1}$) поворот в момент времени t_i руля морского судна, находящегося в точке d_i , приводит к перемещению судна за следующий отрезок времени Δt на расстояние P_i по направлению этой нормированной силы тяги. Одновременно судно продолжает двигаться в прежнем направлении, проходя за время Δt расстояние $\Delta d_i - B\Delta d_i |\Delta d_i|$. Суммарное приращение расстояния Δd_{i+1} в точке

d_{i+1} определяется векторным сложением обоих приращений, пропорциональных составляющим скорости движения (рис. 37).

Направление силы тяги удобно определять относительно прямоугольных координат x и y . В этом случае уравнение движения представляется двумя расчетными формулами $x_{i+1} = x_i + \Delta x_i + P_i \cos \varphi_i - B\Delta x_i |\Delta x_i|$; $y_{i+1} = y_i + \Delta y_i + P_i \sin \varphi_i - B\Delta y_i |\Delta y_i|$, где φ — курсовой угол между положительным направлением оси x и направлением силы тяги, отсчитываемый против часовой стрелки с положительным знаком.

При моделировании на микро-ЭВМ по этим формулам движения морских судов достаточно большого водоизмещения целесообразно принимать $\Delta t = 1$ мин; расстояние определять в морских узлах (миля в час). Для этого следует нормировать силу тяги по формуле $P_{\max} = F_{\max} (60)^2 / 1852$ т, а коэффициент сопротивления воды вычислить по формуле $B = P_{\max}/V_{\max}^2$, где V_{\max} определена с размерностью миля/мин. В программе, реализующей вычисления по составным формулам, целесообразно также предупредить (как в программе 52) воз-

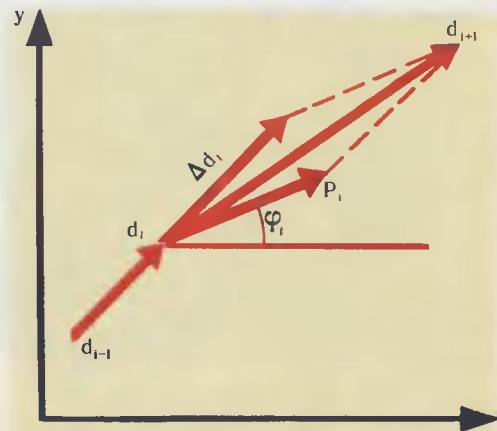


Рис. 37.

Программа 52. Игра «высшие мореходные курсы»

КИП4 Г4	ИП3 63	ПП 53	26 26	\sin 1C	\times 12	$+$ 10	П3 43	ИП2 62	$+$ 10
П2 42	ИП6 66	ПП 53	26 26	\cos 1Г	\times 12	$+$ 10	П6 46	ИП5 65	$+$ 10
П5 45	Л0 5Г	00 00	С/П 50	БП 51	00 00	\uparrow 0Е	\uparrow 0Е	x^2 22	$\sqrt{ }$ 21
\times 12	ИПВ 6L	\times 12	$-$ 11	ИП9 69	ИП8 68	В/О 52	Cx 0Г	П2 42	П3 43
П4 44	П5 45	П6 46	1 01	П0 40	БП 51	00 00			

Инструкция. (Установить переключатель Р — Г в положение Г, ввести $B = F_{\max} = 3600/V_{\max}^2$ (мили/мин) = РВ; для движения с места с координатами $x_0 = y_0 = 0$ ввести $\varphi_0 = P8$; $P_0 = P9$; нажать клавиши БП 3 7 С/П; для движения с места с произвольными координатами ввести $x_0 = P5$; $y_0 = P2$; $1 = P0$; $0 = P3 = P4 = P6$; нажать клавиши В/О и С/П,

в дальнейшем, сохраняя или изменения $\varphi_i = P8$, $P_i = P9$, нажимать для пуска программы клавиши В/О и С/П или С/П, для вычисления координат судна через n минут перед пуском программы ввести $n = P0$; после каждого i -го выполнения программы $PX = P5 = x$; $PY = P2 = y$; $P6 = \Delta x$; $P3 = \Delta y$ (расстояния в милях), $P4 = i$, мин.

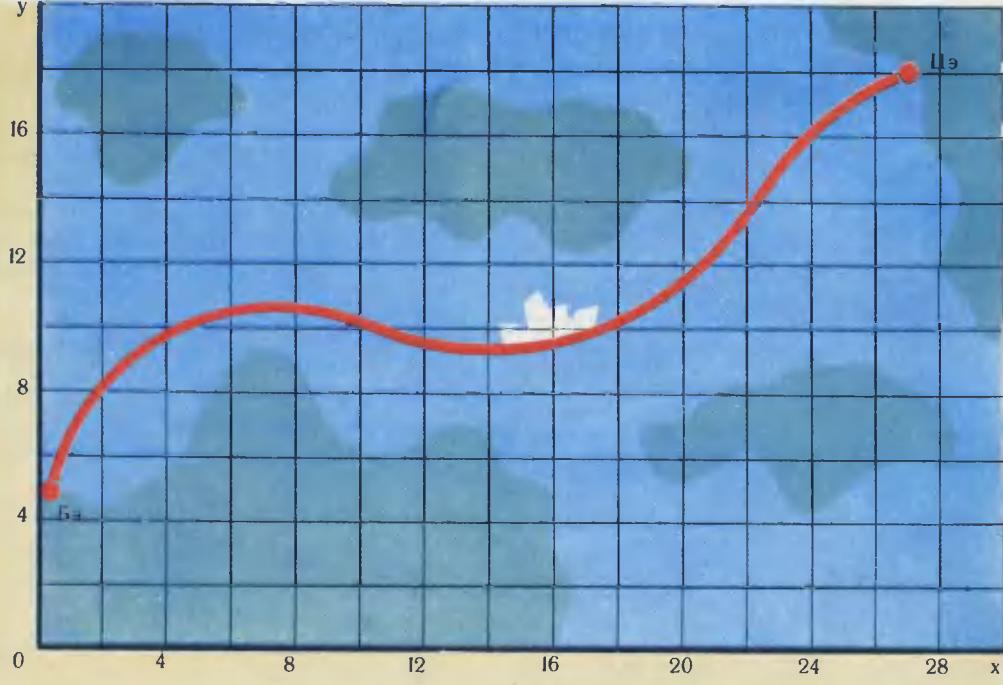


Рис. 38.

можность использования корабельного «автоштурмана» для вычисления координат судна через заданное число n интервалов времени Δt .

Выберем для испытаний судно массой $m = 5000$ т, максимальной силой тяги $F_{\max} = 463\,000$ Н, $\Delta F \leq 115,75$ кг/мин, $\varphi \leq 15^\circ/\text{мин}$ и максимальной скоростью $V_{\max} = 37,5$ морских узлов. Для $\Delta t = 1$ мин находим $P_{\max} = 463 \cdot 60^2 / (1852 \cdot 5000) = 0,18$; $\Delta P \leq 0,045$ и $B = P_{\max} / V_{\max}^2 / 60^2 = 1,0368$.

Проведем ходовые испытания такого судна по программе 52 для $x_0 = y_0 = 0$, приняв $\varphi_0 = 45$ и $P_i = 0,045$; $0,09$; $0,135$; $0,18$ и получив соответственно $x_{i+1} = y_{i+1} = 0,0318$; $0,1262$; $0,3069$; $0,5809$ и $\Delta x_{i+1} = \Delta y_{i+1} = -0,0318$; $0,094$; $0,1806$; $0,2741$. При $6 = PO$; $\varphi_i^0 = 45$; $P_i = 0,18$; получим $x_{i+1} = y_{i+1} = 2,645$; $\Delta x_{i+1} = \Delta y_{i+1} = -0,3503$. Следовательно, за 10 мин судно прошло расстояние $\Delta d = 2x^2 = 3,74$ мили и достигло скорости $V = \sqrt{2\Delta x^2} = 0,454$ мили в минуту или $0,454 \cdot 60 = 29,72$ морских узла.

При $P_i = 0,18$, приняв $\varphi_i^0 = 30$; 15 ; 0 , соответственно получим $x_{i+1} = 3,024$; $3,428$; $3,842$; $\Delta x_{i+1} = 0,3789$; $0,4093$; $0,4148$; $y_{i+1} = 2,958$; $3,216$; $3,405$; $\Delta y_{i+1} = 0,3131$; $0,258$; $0,189$. Таким образом, при повороте на 45° за 3 мин сохранится значительная инерционная составляющая, что необходимо учитывать капитану судна при его вождении вблизи берега и отмелей.

После проведенных испытаний судна с выбранными характеристиками можно составлять карту акватории (подобную показанной на рис. 38) и отправляться в рейс между выбранными портами Бэ и Цэ, помня, что капитан должен привести судно в порт назначения с минимальными затратами времени на весь рейс, включая и остановку (соответствующую снижению скорости ниже установленного значения) в заданном месте.

При движении колесного транспорта направление приращения расстояния через каждый шаг Δt совпадает с направлением приложенной силы тяги, определяемым положением колес автомобиля или железнодорожных рельсов для поезда. В этом случае приращение расстояния Δd будет равно сумме P_i и проекции инерционной составляющей на направление силы тяги. Координаты достигнутой точки в этом случае определяются следующими формулами (рис. 39):

$$x_{i+1} = x_i + ((\Delta d_i - B\Delta d_i |\Delta d_i|) \cos \psi_i + P_i) \cos \psi_i;$$
$$y_{i+1} = y_i + ((\Delta d_i - B\Delta d_i |\Delta d_i|) \cos \psi_i + P_i) \sin \psi_i,$$

где угол изменения направления (угол поворота за Δt) $\psi_i = \varphi_i - \varphi_{i-1}$ и приращение расстояния $\Delta d_i = \sqrt{\Delta x_i^2 + \Delta y_i^2}$.

Так как направление приращения Δd определяется углом ψ_i , то при вычислениях можно принять $B\Delta d |\Delta d| = B(\Delta d)^2$. Для торможения моделируемого автомобиля можно воспользоваться алгоритмом, реализованным в программе 51.

При повороте автомобиля возникает центробежная инерционная составляющая сил, численно равная $Q = -[\Delta d_i - B(\Delta d_i)^2] \sin \psi_i$, которая может привести при крутых поворотах на большой скорости к заносу (боковому скольжению) и даже опрокидыванию автомобиля (как и любого другого колесного самоходного средства). Критическое значение Q_{kp} этой силы для автомобиля определяется опрокидывающим моментом $M = 2h/l$, где h — высота центра тяжести, а l — ширина колеи. Поэтому при моделировании движения автомобиля следует учсть влияние критического значения центробежной инерционной составляющей.

Эти соображения учтены при составлении программы 53, достаточно точно моделирующей движение автомобиля в различных условиях. Как

Программа 53. Игра «вождение автомобиля»

ИП8 68	ИП7 67	+	10	П7 47	КИП4 Г4	ИПД 6Г	$x = 0$ 5Е	29	29	ИПЗ 63	x^2 22
ИП6 66	x^2 22	+	10	$\sqrt{ } 21$	$\uparrow 0E$	x^2 22	ИПВ 6L	\times	12	- 11	П1 41
ИП8 68	\sin 1C	\times	12	ИПА 6—	ХY 14	- 11	$x < 0$ 5C	29	29	$\sqrt{ } 21$	ИП1 61
ИПД 6Г	- 11	$x < 0$ 5C	35 35	Сx 0Г	ИП8 68	\cos 1Г	\times	12	ИП9 69	+	10
П1 41	ИП7 67	\sin 1C	\times	П3 43	ИП2 62	+	10	П2 42	ИП1 61	ИП7 67	
cos 1Г	\times	П6 46	ИП5 65	+	10	П5 45	ИП6 66	С/П 50	БП 51	00 00	
Сx 0Г	П2 42	П3 43	П4 44	П5 45	П6 46	П7 47	ПД 48	БП 51	00 00		
Cx 0Г	5 05	ПД 4Г	БП 51	00 00							

Инструкция. (Установить переключатель Р — Г в положение Г; ввести $Q_{kp} = PA$; $B = PB$); ввести $\psi_0 = P8$; $P_0 = P9$; при старте с места с координатами $x_0 = y_0 = 0$ и $\varphi_0 = 0$ для первого пуска программы нажать клавиши БП 6 0 С/П, при старте с места с другими координатами ввести $x_0 = P5$; $y_0 = P2$; $\varphi_0 = -P7$; $0 = P3 = P4 = P6$ и для первого пуска программы нажать клавиши В/О и С/П; перед следующими пусками программы изменить

и при движении кораблей, скорость автомобиля соответствует приращению расстояния $\Delta d_i = \sqrt{\Delta x_i^2 + \Delta y_i^2}$ за время Δt .

Выберем для испытаний автофургон массой $m = 1200$ кг, максимальной силой тяги $F_{max} = 3600$ Н, допустимыми измерениями силы тяги $\Delta F \leq 900$ Н/с, максимальным углом поворота $\psi = 15^\circ/C$, максимальной скоростью $V_{max} = 144$ км/ч, высотой центра тяжести 1,5 м и шириной колеи 2 м.

При $\Delta t = 1$ с вычислим $P_{max} = F_{max}/m = 3$; $\Delta P \leq 0,75/c$; $V_{max} = 40$ м/с; $B = P_{max}/V_{max}^2 = 1,875 \times 10^{-3}$ и $Q_{kp} = 2 \cdot 1,5/2 = 1,5$.

Для проверки характеристик движения по прямой при $\psi = 0$ вводим

или сохранить угол поворота руля $\psi_i = P8$ и нормированную силу тяги $P_i = P9$ и нажать клавиши В/О и С/П или С/П; для торможения, уменьшая силу тяги до нуля, нажать клавиши БП 7 0 С/П и перед следующими пусками клавишу С/П; после каждого выполнения программы $PX = P6 = \Delta x_i$; $PY = P5 = x_i$; $P2 = y_i$; $P3 = \Delta y_i$; $P4 = i$; $P7 = \varphi_i$; $P8 = \psi_i$; $P9 = P_i$; при опрокидывании автомобиля высвечивается символ ЕГГОГ.

$P_i = 0,75; 1,5; 2,25; 3; 3; \dots$, пустив первый раз программу нажатием клавиш БП 6 0 С/П и получив: $\Delta x_i = 0,75; 2,25; 4,49; 7,45; 10,35; \dots$; $x_i = 0,75; 3; 7,49; 14,94; 25,29; \dots$ Продолжив моделировать нажатием клавиши С/П, установим, что моделируемый автомобиль через 10 с достигнет скорости 22,92 м/с или 82,5 км/ч и пройдет 136,2 м, а через 38 с достигнет практически предельной скорости 39,76 м/с, пройдя расстояние 1125,4 м. Записав содержимое регистров памяти, введем $\psi = 15^\circ$, проверив выполнение программы нажатием клавиши ПП, установим, что в этом случае при повороте на максимальной скорости возникнет центробежная сила $Q = 9,52$, прерывающая

Q_{kp} . Следовательно, при такой скорости поворот автомобиля на большой угол ψ вызовет опрокидывание. Восстановив содержимое регистров памяти до поворота, примем $\psi = 0$; $P_i = 2,25$ и, пустив программу нажатием клавиш БП 7 0 С/П, перед дальнейшими пусками программы нажатием клавиши С/П уменьшим тягу до нуля. Испытание свидетельствует, что моделируемый автомобиль полностью тормозится при максимальной скорости за 9 с при тормозном расстоянии 113,6 м.

Проведя испытание выбранного автомобиля, можно составить карту маршрута с указанием границ дорог и препятствий и отправиться в путешествие.

С помощью микро-ЭВМ можно успешно моделировать движение инерционных объектов также в вертикальной плоскости. Для примера рассмотрим моделирование движения управляемого подводного аппарата (подводной лодки или батискафа) массой m_0 (равной массе вытесненной воды) в уравновешенном состоянии при наполовину заполненных балластных цистернах. При полном заполнении этих цистерн масса аппара-

та $m = m_0 + \Delta m$, и он начинает погружаться под воздействием силы $g\Delta m$ с возрастающей скоростью, ограниченной сопротивлением воды. Давление на аппарат возрастает и при глубине, превышающей критическую величину y_{kp} , возникает опасность разрушения аппарата. При продувке балластных цистерн $m = m_0 - \Delta m$, и аппарат всплывает под воздействием подъемной силы $-g\Delta m$.

Движение аппарата в горизонтальной плоскости и частично в вертикальной обеспечивается движителем с максимальной силой тяги F_{max} , направление которой может изменяться в пределах $|\varphi| \leq \varphi_{max}$ относительно горизонтали (рис. 40). С учетом различного сопротивления воды (зависящего от формы аппарата) при движении по горизонтали и вертикали текущие координаты подводного аппарата определяются формулами

$$x_{i+1} = x_i + \Delta x_i + (F_i \cos \varphi_i - A_1 \Delta x_i |\Delta x_i|)/m;$$

$$y_{i+1} = y_i + \Delta y_i + (F \sin \varphi_i - A_2 \Delta y_i |\Delta y_i| - d)/m,$$

где $A_1 = (\Delta t)^2 F_{max} / (\Delta x_{max})^2$; $A^2 = d \times (\Delta y_{max})^2$; $d = 9,81(\Delta t)^2 \Delta m$, причем $\Delta m = 0$ в уравновешенном состоянии

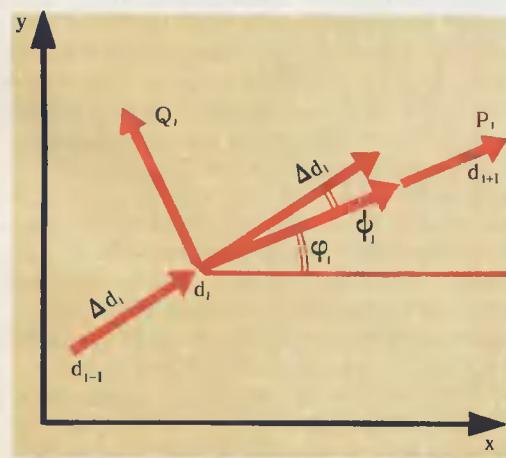


Рис. 39.

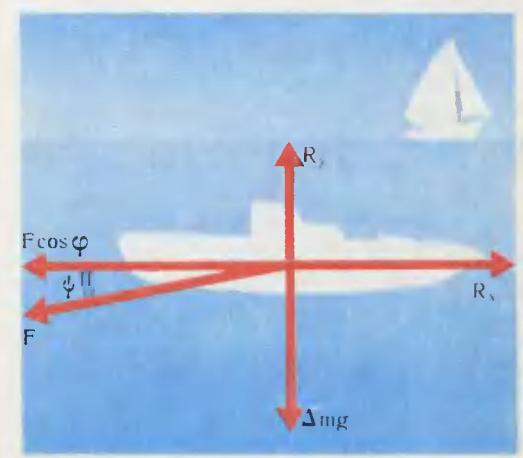


Рис. 40.

Программа 54. Игра «подводное путешествие»

КИП4 Г4	ИП7 67	ИПО 60	+	10	П0 40	ИП9 69	ИП8 68	cos 1Г	×	12	ИП6 66
↑ 0Е	x^2 22	$\sqrt{ }$ 21	\times 12	ИПА 6—	\times 12	— 11	ИП0 60	\div 13	ИП6 66		
+	10	П6 46	ИП5 65	+	10	П5 45	ИП9 69	ИП8 68	sin 1С	\times 12	ИП3 63
↑ 0Е	x^2 22	$\sqrt{ }$ 21	\times 12	ИПВ 6L	\times 12	— 11	ИП1 61	ИП7 67	\times 12		
+	10	ИП0 60	\div 13	ИП3 63	+	10	П3 43	ИП2 62	+	10	$x < 0$ 5С
Сх 0Г	П2 42	ИПС 6С	XY 14	— 11	$x < 0$ 5С	58 58	$\sqrt{ }$ 21	ИП5 65	ИП2 62		
С/П 50	БП 51	00 00	Сх 0Г	П2 42	П3 43	П4 44	П5 45	П6 46	БП		51
00 00											

Инструкция. (Установить переключатель Р — Г в положение Г; ввести $m_0 = P0$; $(\Delta t)^2 9,81 = P1$; $A_1 = PA$; $A_2 = PB$; $y_{kp} = PC$); для первого пуска программы нажимать клавиши БП 6 3 С/П, для последующих — клавиши В/О и С/П или С/П; перед пуском программы при необходимости изменить состояние балластных цистерн, вводя $\Delta m > 0$, $\Delta m =$

аппарата; $\Delta m > 0$ при заполнении и $\Delta m < 0$ при продувке балластных цистерн.

Движение подводного аппарата согласно этим формулам моделируется программой 54, при составлении которой предусмотрена сигнализация аварийного состояния после достижения критической глубины.

Выберем для путешествия под водой батискаф массой $m_0 = 10\ 000$ кг (в уравновешенном состоянии); полной емкостью балластных цистерн $2\Delta m = 500$ кг; максимальной силой тяги двигателя $F_{max} = 500$ Н (при заднем ходе $F < 0$) с допустимыми изменениями $\Delta F \leq 100$ Н/с и направления силы тяги на угол $\varphi < 20^\circ$ вверх и вниз от горизонтали; максимальной горизонтальной скоро-

=0 или $\Delta m < 0$ (соответственно при полной и половинной загрузке цистерн и при их продувке) в регистр 7, а также $\varphi_0 = P8$ или $F_i = P9$; после каждого выполнения программы $PX = P2 = y$; $PY = P5 = x$; $P3 = \Delta y$; $P4 = i$; $P6 = \Delta x_i$; при аварийном состоянии ($y > y_{kp}$) высвечивается сигнал ЕГТОГ.

стью $V_{x_{max}} = 5$ м/с; максимальной скоростью свободного погружения $V_{h_{max}} = 3$ м/с и максимальной глубиной погружения $y_{kp} = 100$ м.

При $\Delta t = 1$ с находим $A_1 = F_{max}/(\Delta x_{max})^2 = 20$; $A_2 = 9,81\Delta m/(\Delta y_{max})^2 = 272,5$. Приняв $\Delta m = 250$; $\varphi_0 = 0$; $F_0 = 0$, с помощью программы 54 определим, что через 10 с батискаф погрузится до глубины $y_{20} = 11,4$ м со скоростью 1,89 м/с. Моделируем уравновешивание батискафа, приняв $\Delta m = 0$, и определим, что еще через 10 с батискаф по инерции достигнет глубины 26,8 м при скорости погружения 1,32 м/с.

Моделируем включение двигателя и набор максимальной мощности за 5 с при $\varphi = 0$ и определим, что через 10 с батискаф, продолжая погружать-

ся по инерции, достигнет глубины 38,26 м при скорости погружения 1,02 м/с и пройдет по горизонтали 1,34 м от места погружения со скоростью 0,3 м/с. Моделируем продувку балластных цистерн, приняв $\Delta t = -250$, и определим, что через 5 с батискаф достигнет по инерции максимальной глубины $y_{25} = 40$ м, после чего начнет всплывать, достигнув поверхности воды через 56 с с вертикальной скоростью 1,74 м/с на расстоянии 26,5 м от места погружения. Батискаф испытан и теперь, запасшись картой подводного рельефа, можно с помощью программы 54 отправляться в подводное путешествие.

Подобным образом можно моделировать перемещение в вертикальной плоскости и других инерционных объектов, учитывая особенности их движения. Моделирование движения таких объектов несколько упрощается при отсутствии сопротивления среды, что характерно для космических полетов за пределами атмосферы. В качестве простейшего примера рассмотрим управление тормозным двигателем для мягкой посадки на Луну космической станции, радиально приближающейся к ней со скоростью V_0 на высоте h_0 (рис. 41). Движение такой станции описывается простым уравнением $ma = mg - F$, где m и a — масса и ускорение станции; $g = g_0/(1 + h/r)^2$ — ускорение тяготения, равное $g_0 = 1,64$ м/с² на поверхности Луны с радиусом $r = 1738$ м, $F = C\Delta m$ — сила тяги, определяемая расходом топлива Δm за единицу времени.

Заменив a разностным отношением $\Delta V/\Delta t$, составим расчетную формулу $h_{i+1} = h_i + \Delta t(V_i - g_0/(1 + h_i/r)^2 + c\Delta m_i/m_i)$, где C — коэффициент пропорциональности, зависящий от конструкции тормозного двигателя; $m_i = m_0 + m_{ti}$ — Δm_i , где m_0 — масса основной

части станции; m_{ti} — масса топлива. Для «мягкой» посадки с минимальной скоростью при приближении к поверхности Луны приходится достаточно быстро принимать решения, тогда как при больших расстояниях от Луны для ускорения вычислений целесообразно выбирать большой шаг Δt . Поэтому при составлении программы 55, моделирующей управление посадкой космической станции, учтена возможность использования переменного шага $\Delta t = 1\dots10$ с. При больших значениях шага в этом случае возникает погрешность в определении координат, но ее можно «объяснить» более равномерной работой двигателя при $\Delta t > 1$ с, так как для игровых задач эта погрешность несущественна. Так как скорость станции определяется отрицательным приращением высоты, то ей следует приписывать отрицательный знак.

Цель игры с программой 55 заключается в посадке станции на поверхность Луны с достаточно малой скоростью при сохранении достаточного запаса топлива. Так, при $m_0 = 11\ 000$ кг; $m_{t0} = 9000$ кг; $h_0 = 1000$ км; $V_0 = -5$ км/с и $C =$

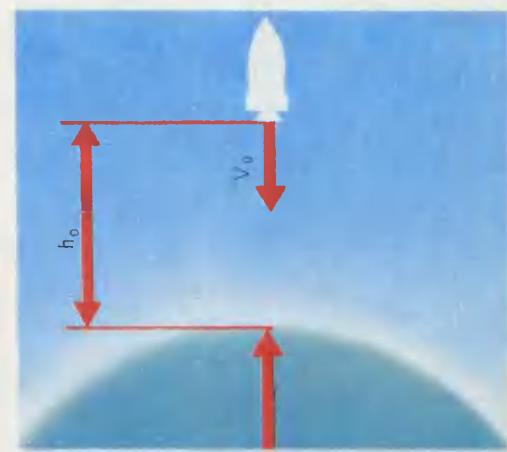


Рис. 41.

= 120 рациональное расходование топлива обеспечивает посадку со скоростью менее 1 м/с, но преждевременный расход топлива приводит к аварийному падению станции на поверхность Луны с большой скоростью.

Более сложно управлять посадкой космической станции, находящейся на орбите или приближающейся к месту посадки под углом относительно радиального направления, так как в этом случае приходится учитывать сложные гравитационные зависимости. Вместе с этим при полетах ракет на небольшие расстояния, когда можно пренебречь кривизной земной поверхности и влиянием ее вращения, моделирование упрощается. Однако в этом случае приходится дополнительно учитывать сопротивление воздуха в нижних слоях атмосферы, плотность которого изменяется с высотой пропорционально множителю $e^{-1,16 \cdot 10^{-4} h}$. При этом полет ракеты массой $m = m_0 + m_t$, где m_t — масса топлива, описывается формулами (рис. 42)

$$x_{i+1} = x_i + \Delta x_i + (\Delta t)^2 \cos \varphi [C \Delta m_i - B e^{A h_i} (\Delta d_i)^2 / m_i];$$

$$h_{i+1} = h_i + \Delta h_i + (\Delta t)^2 \sin \varphi [C \Delta m_i -$$

$- B e^{A h_i} (\Delta d_i)^2 / m_i - g]$, где $m_i = m_0 + m_{t,i-1} - \Delta m_i$ — текущая масса ракеты; Δm — расход топлива за 1 с; C — коэффициент силы тяги; B — коэффициент силы сопротивления воздуха; $A = -1,16 \cdot 10^{-4} \text{ м}^{-1}$;

$$\Delta d = \sqrt{\Delta x^2 + \Delta h^2}; g = 9,81 \text{ м/с}^2; \varphi — \text{угол направления силы тяги относительно горизонтали.}$$

Коэффициент C определяется конструкцией двигателя: $B = C \Delta m / (\Delta d_{\max})^2$ — максимальным приращением расстояния в нижних слоях атмосферы при максимальной тяге. Движение ракеты согласно приведенным формулам при $\Delta t = 1$ с моделируется программой 56.

Простейшая цель игры с программой 56 может заключаться в управлении полетом ракеты на заданное расстояние за минимальное время. Так как сопротивление верхних слоев атмосферы невелико, то эта цель достигается при рациональном выборе траектории полета ракеты. Для примера выберем характеристики ракеты с относительно маломощным и неэкономичным двигателем, характерным для наиболее ранних конструкций: $m_0 = 1000$ кг; $m_{t0} = 1000$ кг; $\Delta m = 10$ кг/с; $C = 2000$; максимальная

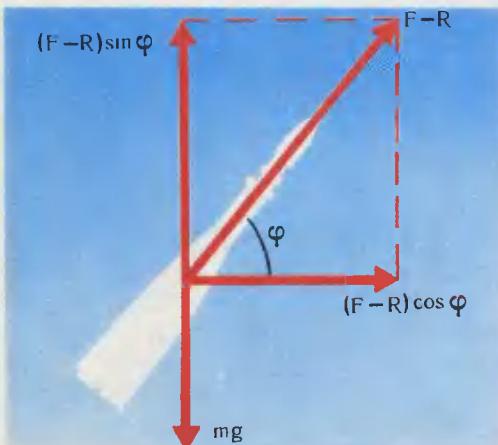


Рис. 42.

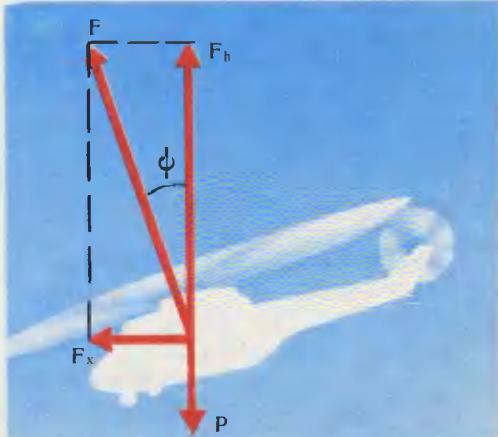


Рис. 43.

Программа 55. Игра «посадка на Луну»

ИП0 60	ИП4 64	+	10	П4 44	ИПС 6C	ИП9 69	\times	12	ИП6 66	ИП9 69	ИП0 60
\times 12	— 11	$x < 0$ 5C	16 16	Сх 0Г	П9 49	П6 46	ИП5 65	+	10	\div	13
ИПА 6—	ИП0 60	\times	12	ИП7 67	ИПВ 6L	\div	13	1 01	+	x^2 22	\div 13
— 11	ИП8 68	+	10	П1 41	ИП0 60	\times	12	П8 48	ИП7 67	+	10 П7 47
ИП1 61	С/П 50	БП 51	00 00	Сх 0Г	П4 44	1 01	,	0—	6 06	4 04	
ВП 0C	3 03	/—/ 01	ПА 4—	1 01	7 07	3 03	8 08	ПВ 4L	БП 51		
00 00											

Инструкция. ($C = PC$); $h_0(\text{км}/\text{с}) = P7$; $-V_0(\text{км}/\text{с}) = P8$; $m_0 = P5$; $m_i = P6$ перед каждым пуском программы вводить или сохранять $\Delta t = P0$ ($1 \leq \Delta t \leq 10$); $\Delta m_i = P9$; для первого пуска нажать клавиши БП44С/П, для послед-

ующих — клавиши В/О и С/П или С/П; после каждого выполнения программы $PX = P1 = -V_1(\text{км}/\text{с})$; $PY = P7 = h_i(\text{км})$; $P4 = i$; $P6 = m_{ti}(\text{кг})$.

Программа 56. Игра «управление ракетой»

КИП4 Г4	ИПС 6C	ИП9 69	\times	12	ИП6 66	x^2 22	ИП3 63	x^2 22	+	10	ИПВ 6L
\times 12	ИПА 6—	ИП2 62	\times	12	e^x 16	\times	12	— 11	ИПД 6Г	ИП9 69	— 11
$x < 0$ 5C	24 24	Сх 0Г	ПД 4Г	ИП7 67	+	10	\div 13	\uparrow 0Е	ИП8 68	cos 1Г	
\times 12	ИП6 66	+	10	П6 46	ИП6 66	+	10	П5 45	ХУ 14	ИП8 68	sin 1C
\times 12	ИП1 61	— 11	ИП3 63	+	10	П3 43	ИП2 62	+	10	П2 42	L0 5Г
00 00	С/П 50	БП 51	00 00	Сх 0Г	П2 42	П3 43	П4 44	П5 45	П6 46		
1 01	П0 40	9 09	,	8 0—	1 08	1 01	П1 41	1 01	,	1 0—	
6 06	/—/ 01	ВП 01	4 04	/—/ 01	ПА 4—	БП 51	00 00				

Инструкция. (Установить переключатель Р — Г в положение Г; ввести $B = PB$; $C = PC$; $m_0 = P7$); $m_{t0} = PД$, при необходимости перед пусками программы изменять $q_i = P8$; $\Delta m_i = P9$; для первого пуска программы нажать клавиши БП 5 4 С/П, для последующих

пусков — клавиши В/О и С/П; после каждого выполнения программы $PX = P2 = h_i$; $PY = P5 = x_i$; $P3 = \Delta h$; $P4 = i(c)$; $P6 = \Delta x$; $PД = m_{ti}$; для вычисления координат через $n > 1$ шагов $\Delta t = 1$ с вводить $n = P0$.

скорость в нижних слоях атмосферы $V_{\max} = 1080$ км/ч = 300 м/с; $\Delta\varphi \leqslant 10^\circ/\text{с}$. Следовательно, $B = 2000 \times 10/300^2 = 2,222\ 222\ 2 \cdot 10^{-1}$.

Приняв $\varphi = 90^\circ$; $\Delta t = 10$ кг, с помощью программы 56 найдем, что через 1 с ракета поднимется по вертикали на 0,24 м, а через 50 с ракета достигнет высоты 1513,4 м со скоростью 86,4 м/с. Уменьшим с допустимой скоростью за 4 с угол φ до 50° , после чего ракета достигнет высоты 1880,5 м со скоростью 92,7 м/с и переместится по горизонтали на расстояние 43 м со скоростью 21 м/с. Продолжив испытания при $\varphi = 50^\circ$ и $\Delta t = 10$ кг/с, определим, что после сгорания топлива через 100 с ракета продолжит двигаться по инерции и через 105 с после старта достигнет максимальной высоты 5277 м (при вертикальной скорости 0,53 м/с) на расстоянии 11 187 м от места старта, разогнавшись до скорости 355,8 м/с по горизонтали. Продолжая моделирование, установим, что ракета упадет на землю через 147 с после старта на расстоянии 22 874 м с вертикальной скоростью 452,9 м/с и горизонтальной скоростью 246,9 м/с. Выбрав более мощную ракету, можно моделировать ее полеты на большие расстояния, используя топливо как для разгона, так и для точного приземления ракеты.

При полетах ракет сила земного тяготения преодолевается только вертикальной составляющей силы тяги. Подобная особенность характерна и для полета вертолетов, горизонтальные перемещения которых обеспечиваются наклоном винта на некоторый угол ψ (рис. 43). С учетом изменения плотности воздуха при наборе высоты полет вертолета моделируется формулами

$$x_{i+1} = x_i + \Delta x_i + e^{Ah_i} (P_i \sin \psi_i - B \Delta x_i |\Delta x_i|);$$

$$h_{i+1} = h_i + \Delta h_i + e^{Ah_i} (P_i \cos \psi_i - C \Delta h_i |\Delta h_i|) - d,$$

где $P_{\max} = F_{\max} (\Delta t)^2 / m$; $B = P_{\max} / (\Delta x_{\max})^2$; $C = (P_{\max} - d) / (\Delta h_{\max})^2$; $A = -1,16 \cdot 10^{-4/\text{м}}$, $d = 9,81 (\Delta t)^2$.

В этих соотношениях, моделируемых программой 57, пренебрегали изменением массы m вертолета по мере сгорания топлива, что допустимо при полетах на небольшие расстояния.

Пусть моделируется полет вертолета массой $m = 2000$ кг, максимальной силой тяги у земли $F_{\max} = 25\ 000$ Н, $\Delta F \leqslant 5000$ Н/с, предельным углом наклона винта $\varphi \leqslant 10^\circ$ и $\Delta\varphi \leqslant 2^\circ/\text{с}$, максимальной горизонтальной скоростью $V_{x\max} = 180$ км/ч и предельной вертикальной скоростью $V_{z\max} = 27$ км/ч и при допустимой вертикальной скорости посадки $V_z \leqslant 2$ м/с.

Приняв $\Delta t = 1$ с, вычислим $P_{\max} = 12,5$; $\Delta P \leqslant 2,5/\text{с}$; $C = (12,5 - 9,81)/7,5^2 = 4,782222 \cdot 10^{-2}$; $B = 12,5 \sin 10^\circ / 50^2 = 8,6824084 \times 10^{-4}$. При $\varphi = 0$, как следует из испытаний по программе 56, и допустимом увеличении силы тяги вертолет через 541 с достигнет максимальной высоты 2102 м. При увеличении угла наклона до $\varphi = 10^\circ$ через 545 с с момента старта вертолет пройдет в горизонтальном направлении 3477,7 м, достигнув скорости $V_x = 49,2$ м/с при снижении до высоты 1974,5 м со скоростью $V_z = -0,4$ м/с. Дальнейшие испытания показывают, что вертолет хорошо управляемся в обоих направлениях оси x (при $P > 0$ и $P < 0$), но посадка вертолета с допустимой посадочной скоростью требует особого внимания пилота, так как инерционность вертолета достаточно большая.

Несложно моделировать движение в воде или в воздухе и для трех измерений, дополнив составленные

Программа 57. Игра «испытатели вертолетов»

ИП9 69	ИП8 68	\cos 1Г	\times 12	ИПС 6C	ИПЗ 63	\uparrow 0Е	x^2 22	$\sqrt{ }$ 21	\times 12
\times 12	— 11	ИПА 6—	ИП2 62	\times 12	e^x 16	П1 41	\times 12	ИПД 6Г	— 11
ИПЗ 63	+	П3 43	ИП2 62	+	10	П2 42	КИП4 Г4	ИП9 69	ИП8 68
\times 12	ИПВ 6L	ИП6 66	\uparrow 0Е	x^2 22	$\sqrt{ }$ 21	\times 12	\times 12	— 11	ИП1 61
\times 12	ИП6 66	+	10	П6 46	ИП5 65	+	10	П5 45	Л0 5Г
C/П 50	БП 51	00 00	Сх 0Г	П2 42	П3 43	П4 44	П5 45	П6 46	1 01
П0 40	БП 51	00 00							

Инструкция. (Установить переключатель Р — Г в положение Г, ввести $A = -1,16 \times 10^{-4} = PA$; $B = PB$; $C = PC$; $(\Delta t)^2 9,81 = PД$); для первого пуска программы нажать клавиши БП 5 3 С/П; для последующих — клавиши В/О и С/П или С/П; перед пуском программы

изменить при необходимости $\psi_i = P8$; $P_i = P9$; для вычисления координат через n интервалов Δt ввести (при $n > 1$) $n = P0$; после каждого i -го выполнения программы (время счета около 18 с на один шаг) $PX = P2 = h_i$; $PY = P5 = x_i$; $P3 = \Delta h_i$; $P4 = i$; $P6 = \Delta x_i$.

Программа 58. Игра «гигантский слалом»

П9 49	ИП7 67	+	10	П7 47	КИП4 Г4	ИПЗ 63	x^2 22	ИП6 66	x^2 22	+	10
П1 41	$\sqrt{ }$ 21	ИП1 61	ИПВ 6L	\times 12	— 11	ИП9 69	\cos 1Г	\times 12	ИПА 6—		
ИП7 67	\cos 1Г	\times 12	+	10	П1 41	ИП7 67	\sin 1С	\times 12	П6 46	ИП5 65	
+	П5 45	ИП1 61	ИП7 67	\cos 1Г	\times 12	П3 43	ИП2 62	+	10	П2 42	
C/П 50	БП 51	00 00	\uparrow 0Е	Сх 0Г	П2 42	П3 43	П4 44	П5 45	П6 46		
П7 47	\rightarrow 25	БП 51	00 00	П8 48	ИП2 62	ИП5 65	ИП8 68	— 11	ИПЗ 63		
ИП6 66	\div 13	\times 12	— 11	БП 51	40 40						

Инструкция. ($9,81 \cos \alpha = PA$; $B = PB$) перед каждым пуском программы вводить угол поворота лыж $\psi_i = PX$; для первого пуска нажать клавиши БП 4 3 С/П; для последующих — клавиши В/О и С/П или С/П; после каждого выполнения программы $PX = P2 =$

$= y_i$; $PY = P5 = x_i$; $P3 = \Delta Y_i$; $P4 = i$; $P6 = \Delta x_i$; $P7 = \varphi_i$; для определения координаты $y_{\text{пер}}$ точки пересечения вертикальной линии с координатой x_{φ} ввести $x_{\varphi} = PX$ и нажать клавиши БП 5 4 С/П, что приведет к высвечиванию $y_{\text{пер}}$.

расчетные формулы и программы фрагментами для вычисления третьей координаты. Однако время выполнения программы в этом случае заметно возрастет, что при использовании программируемых микрокалькуляторов с относительно малым быстрым действием снизит игровой эффект. Следует добавить, что моделирование поведения реальных объектов на ЭВМ является одним из наиболее важных для практики применения быстро действующих вычислительных машин. В некоторых случаях движение в третьем измерении приходится учитывать косвенно. Примером может служить спуск лыжника по плоскому склону горы, отклоненному от вертикали на угол α . В этом случае при движении лыжника массой m вдоль линии максимального уклона (которую целесообразно выбрать в качестве оси координат y с положительным направлением в сторону спуска) на него действует сила тяги $F = mg \cos \alpha = mg_1$. Если лыжи направлены под углом φ к этой линии, то на лыжника действует сила тяги $mg_1 \cos \varphi$ с проекциями $F_x = mg_1 (\cos \varphi)^2$ и $F_y = mg_1 \cos \varphi \sin \varphi$ на координатные оси (рис. 44). Максимальная скон

рость лыжника V_{\max} ограничена сопротивлением воздуха и трением лыж о снег, примерно пропорциональным квадрату скорости.

При повороте лыж на угол ψ за время Δt и отсутствии бокового скольжения инерционная составляющая, как и при поворотах колесного транспорта (см. рис. 39), проектируется на направление движения лыж, а центробежную силу лыжник преодолевает наклоном туловища и лыж. Следовательно, спуск лыжника с горы описывается формулами

$$x_{i+1} = x_i + x_i + ((\Delta d_i - B(\Delta d_i)^2) \cos \psi_i + g_1 \cos \varphi_i) \sin \varphi_i;$$

$$y_{i+1} = y_i + \Delta y_i + ((\Delta d_i - B(\Delta d_i)^2) \times \cos \psi_i + g_1 \cos \varphi_i) \cos \varphi_i,$$

где $\Delta d_i = \sqrt{\Delta x_i^2 + \Delta y_i^2}$; $B = g_1 / (\Delta d_i)^2$, φ_i — угол между направлением лыж и положительным направлением оси y ; $\psi_i = \varphi_i - \varphi_{i-1}$ — угол поворота за время Δt (знаки углов совпадают со знаком координаты x в том же направлении); $g_1 = g \cos \alpha$.

Движение лыжника, описываемое этими формулами, моделируется программой 58. Ее удобно использовать для моделирования различных лыжных состязаний, например по слалому. В таких состязаниях лыжники проходят ряд «ворот», обозначенных флагками. Каждая пара флагков с координатой x_f имеет различные координаты y_{f1} и y_{f2} . В программе 58 предусмотрена возможность точного определения координаты $y_{\text{пер}}$ точки пересечения линии флагков (или другой линии вертикальной) по хранящимся в памяти данным согласно формуле $y_{\text{пер}} = y_i - (x_i - x_f) \Delta y_i / \Delta x_i$.

Для тренировок по слалому выберем $\alpha = 60^\circ$, разметим координатную сетку с началом координат в точке старта и разместим пару ворот с координатами флагков $x_1 = 0$; $y_{11} = 150$ м; $y_{12} = 160$ м; $x_2 = -100$ м; $y_{21} = 400$ м; $y_{22} = 410$ м и финишные

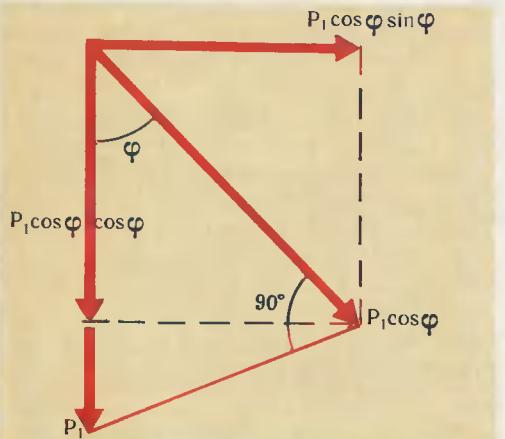


Рис. 44.

ворота с координатами флагов $x_{\phi} = 150$ м; $y_{\phi 1} = 490$ м; $y_{\phi 2} = 500$ м (рис. 45). Приняв $\psi \leqslant 10^\circ$ и $V_{\max} = 72$ км/ч = 20 м/с, при $\Delta t = 1$ с найдем $g_1 = 9,81 \cos 60^\circ = 4,905$ м/с²; $B = 4,905/20^2 = 1,22625 \cdot 10^{-2}$. Полагая флагок сбитым, если линия ворот пересечена ближе 0,4 м от флагка, начнем тренировочные спуски с помощью программы 58, пытаясь пройти без ошибок все ворота за минимальное число шагов.

Относительно сложно моделирование движения крылатых летательных аппаратов, определяемого силой F тяги, направленной перпендикулярно ей подъемной силой Q крыльев, силами R_F и R_Q сопротивления воздуха и силой mg земного тяготения. В прямоугольной системе «самолетных» координат d и r , положительные направления которых совпадают с направлениями сил F и Q , полет самолета под углом φ к горизонту (рис. 46) описывается уравнениями

$$ma_d = F - R_F - mg \sin \varphi;$$

$$ma_r = Q - R_Q - mg \cos \varphi,$$

где a_d и a_r — проекции ускорения самолета на оси координат.

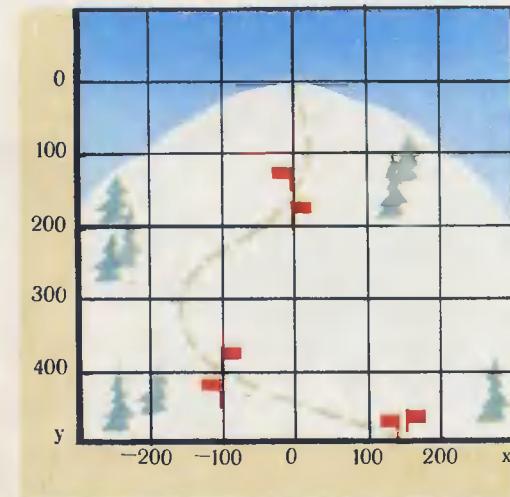


Рис. 45.

Подъемная сила пропорциональна квадрату скорости V_d и зависит от положения закрылков или других управляемых элементов крыльев. При разбеге самолета по взлетной полосе до скорости $V_{\text{вал}}$ подъемная сила компенсирует силу земного тяготения, самолет отрывается от земли и набирает высоту. С увеличением высоты подъемная сила уменьшается пропорционально уменьшению плотности воздуха и на максимальной высоте h_{\max} компенсируется силой земного тяготения. С уменьшением плотности воздуха уменьшается его сопротивление, но одновременно уменьшается и сила тяги большинства авиационных двигателей, в связи с чем можно пренебречь зависимостью скорости V_d от высоты. Поэтому после приближенной замены ускорения разностным отношением текущие координаты самолета определяются по формулам

$$d_{i+1} = d_i + \Delta d_{i+1} = d_i + \Delta d_i - B \Delta d_i |\Delta d_i| + P_i - G \sin \varphi_{i+1};$$

$$r_{i+1} = r_i + \Delta r_{i+1} = r_i + \Delta r_i - D \Delta r_i |\Delta r_i| + C e^{A h_i} (\Delta d_i)^2 - G \cos \varphi_{i+1},$$

где максимальная нормированная сила тяги $P_{\max} = F_{\max}(\Delta t)^2/m$; коэф-

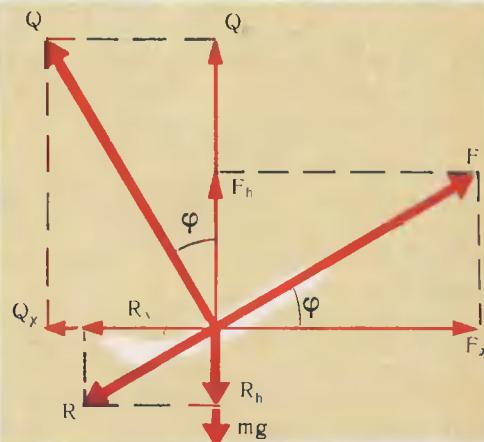


Рис. 46.

фициенты сопротивления воздуха $B = P_{\max}/V_{\max}^2(\Delta t)^2 = P_{\max}/(\Delta d_{\max})^2$ и $D = (C(\Delta d_{\max})^2 - G)/(\Delta r_{\max})^2$; нормированная сила тяготения $G = 9,81(\Delta t)^2$, «взлетный» коэффициент $C = G/(V_{\text{взл}}^2)(\Delta t)^2 = G/(\Delta d_{\text{взл}})^2$, «высотный» коэффициент $A = (1/h_{\max}) \ln (\Delta d_{\text{взл}}/\Delta D_{\max})^2$, а коэффициент q определяет положение закрылков ($q_{\min} \leq q \leq 1$).

Текущие значения «земных» координат x и h (рис. 46) определяются формулами преобразования координат

$$\begin{aligned}x_{i+1} &= x_i + \Delta d_{i+1} \cos \varphi_{i+1} - \\&\quad - \Delta r_{i+1} \sin \varphi_{i+1}; \\h_{i+1} &= h_i + \Delta d_{i+1} \sin \varphi_{i+1} + \\&\quad + \Delta r_{i+1} \cos \varphi_{i+1}.\end{aligned}$$

Вычисление приращений Δd_{i+1} и Δr_{i+1} усложняется при изменении направления силы тяги на угол $\psi_i = \varphi_{i+1} - \varphi_i$ с помощью рулей высоты, обычно управляемых положением рукоятки штурвала. Действие рулей высоты пропорционально квадрату скорости V_d , а положение рукоятки штурвала можно определить коэффициентом m , причем граничным значениям интервала $-1 \leq m \leq 1$ соответствуют предельные положения штурвала «от себя» и «на себя». В простейшем случае зависимость между положениями штурвала и рулей высоты линейна, когда угол $\psi_i = m\psi_{\max}(\Delta d_i/\Delta d_{\max})^2$ примет максимальное значение ψ_{\max} при $m = 1$ и $\Delta d_i = \Delta d_{\max}$. В действительности направление силы тяги изменяется на меньший угол, так как приращения Δd_{i+1} и Δr_{i+1} определяются векторными составляющими, пропорциональными силами.

Рассмотренные соотношения тем точнее описывают полет самолета, чем меньше шаг Δt . При недостаточно малом шаге возможна неустойчивость вычислений, возникающая вследствие обратной связи между вычисляемыми величинами и проявляющаяся в

периодическом или чрезмерно быстрым изменении результатов вычислений, противоречащем физическим условиям полета. Вероятность неустойчивости вычислений снижается при уменьшении шага Δt , но в этом случае увеличивается время моделирования и приходится использовать микро-ЭВМ с достаточно большим быстродействием. В противном случае (например, при использовании программируемых микрокалькуляторов) необходимо так изменить модель, чтобы без заметного ухудшения ее точности устранить неустойчивость вычислений при выборе достаточно большого шага.

Для полетов на малые расстояния допустимо пренебречь изменением массы m самолета при сгорании топлива. Кроме того, при малом шаге можно заменить векторную сумму приращений алгебраической, а для уменьшения неустойчивости вычислений — учитывать силу R_q экспоненциальным множителем. В этом случае

$$\begin{aligned}\Delta d_{i+1} &\approx \Delta d_i - B\Delta d_i|\Delta d_i| + P_i - \\&\quad - G \sin \varphi_{i+1};\end{aligned}$$

$$\begin{aligned}\Delta r_{i+1} &\approx e^{\gamma|\Delta r_{i+1}|} [\Delta r_i + qCe^{Ah_i}(\Delta d_i)^2 - \\&\quad - G \cos \varphi_{i+1}],\end{aligned}$$

где Δr_{i+1} — содержимое квадратных скобок; коэффициент $\gamma = (1/2C \times (\Delta d_{\max})^2) \ln (\Delta r_{\max}/2C(\Delta d_{\max})^2)$.

Приняв $q = 1$ (при необходимости изменение положения закрылков можно учесть уменьшением коэффициента C), составим согласно этим формулам программу 59, достаточно точно моделирующую полет самолета на небольшие расстояния.

Выберем спортивный самолет с $m = 1500$ кг; $F_{\max} = 3000$ Н; $\Delta F \leq 750$ Н/с; $V_{\text{взл}} = 144$ км/ч = 40 м/с, $V_{d\max} = 360$ км/ч = 100 м/с, $V_{r\max} = 36$ км/ч = 10 м/с; $h_{\max} = 6000$ м; $\psi_{\max} = 25^\circ$ /с. Приняв $\Delta t = 1$ с, найдем $P_{\max} = 3000/1500 = 2$; $\Delta P \leq 0,5$; $a = \psi_{\max}/(\Delta d_{\max})^2 = 25/100^2 = 2,5 \times$

Программа 59. Игра «школа высшего пилотажа»

КИП4 Г4	ИП2 62	ИПА 6—	\times 12	e^x 16	ИПС 6С	\times 12	ИП6 66	x^2 22	\times 12
Bx 0	ИП8 68	\times 12	ИП1 61	\times 12	ИП7 67	$+$ 10	П7 47	\cos 1Г	ИП0 60
\times 12	— 11	ИП3 63	$+$ 10	\uparrow 0Е	x^2 22	$\sqrt{ }$ 21	ИПД 6Г	\times 12	e^x 16
\times 12	П3 43	ИП6 66	\uparrow 0Е	\uparrow 0Е	x^2 22	$\sqrt{ }$ 21	ИПВ 6L	\times 12	
— 11	ИП9 69	$+$ 10	ИП0 60	ИП7 67	\sin 1C	\times 12	— 11	П6 46	ИП7 67
\cos 1Г	\times 12	ИП3 63	ИП7 67	\sin 1C	\times 12	— 11	ИП5 65	$+$ 10	П5 45
ИП6 66	ИП7 67	\sin 1C	\times 12	ИП3 63	ИП7 67	\cos 1Г	\times 12	$+$ 10	ИП2 62
$+$ 10	$x < 0$ 5С	75	Cx 0Г	П3 43	П2 42	C/П 50	БП 51	00 00	Cx 0Г
П2 42	П3 43	П4 44	П5 45	П6 46	П7 47	БП 51	00 00		

Инструкция. ($G = 9,81(\Delta t)^2 = P0$; $a = \psi_{\max}/(\Delta d_{\max})^2 = P1$; $A = PA$; $B = PB$; $C = PC$; $\gamma = PD$; перед пуском программы при необходимости изменять $t_i = P8$; $P_i = P9$; для первого пуска нажать клавиши БП 7 8 С/П;

для последующих — клавиши В/О и С/П или С/П; после каждого выполнения программы $PX = P2 = h_{i+1}$; $PY = P5 = x_{i+1}$; $P3 = \Delta r_{i+1}$; $P4 = i$; $P6 = \Delta d_{i+1}$; $P7 = \varphi_{i+1}$.

$$\begin{aligned} &\times 10^{-3}; \quad A = 2 \ln(40/100)/6000 = \\ &= -3,054\ 302\ 5 \cdot 10^{-4}; \quad B = 2/100^2 = \\ &= 2 \cdot 10^{-4}; \quad C = 9,81/40^2 = 6,131\ 25 \times \\ &\times 10^{-3}; \quad \gamma = (\ln(10/2C \cdot 100^2)/2C \times \\ &\times 100^2 = -4,322\ 271\ 2 \cdot 10^{-3}. \end{aligned}$$

С помощью программы 59 находим, что при увеличении тяги до максимума за 4 с самолет, пробежав около 500 м, через 23 с оторвется от земли и полетит со скоростью $V_x = 42,35$ м/с при скорости поъема $V_h = 0,33$ м/с. Через 4 мин после старта самолет летит на высоте 1220 м, преодолев при этом расстояние более 20 км со скоростью $V_x = 99,98$ м/с и поднимаясь со скоростью $V_h = 6,3$ м/с.

Приняв $m = 1$ («взял ручку на себя до упора»), находим, что наш самолет за следующие 35 с совершил «мертвую петлю», выходя из нее при $h \approx 1220$ м;

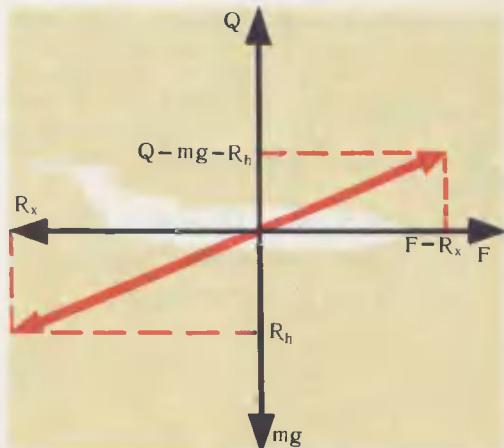


Рис. 47.

Программа 60. Игра «полет на авиалайнере»

КИП4 Г4	ИП9 69	ИП6 66	x^2 22	ИПВ 6L	\times	12	—	11	ИП6 66	+	10	П6 46
ИП5 65	+	10	П5 45	ИПС 6C	ИП6 66	x^2 22	\times	12	ИП8 68	\times	12	ИПА 6—
ИП2 62	\times 12	e^x 16	\times 12	\uparrow 0E	ИПЗ 63	+	10	ИПД 6Г	—	11	ИП1 61	
\times 12	XY 14	\div 13	П3 43	ИП2 62	+	10	$x < 0$ 5C	40 40	Cx 0Г	П3 48		
П2 42	L0 5Г	00 00	ИП6 66	ИП7 67	\times	12	C/П 50	БП 51	00 00	Cx 0Г		
П2 42	П3 43	П4 44	П5 45	П6 46	1	01	П0 40	БП 51	00 00			

Инструкция. ($\Delta h_{\max} = P1$; $A = PA$; $B = PB$; $C = PC$; $9,81 (\Delta t)^2 = PD$; размерный коэффициент скорости $k = P7$); при необходимости изменять $q = 1 = P8$; $P_i = P9$; для вычисления координат через $n > 1$ шагов ввести $n = P0$; для первого пуска программы нажать

$x \approx 20\ 050$ м; $\Delta d \approx 104$ м; $\Delta r = 5,6$ м и $\varphi = 371,8^\circ$. Осторожно уменьшив t , выровняем самолет и выполним следующую фигуру высшего пилотажа или посадку.

Программа 59 обеспечивает достаточно точное моделирование полета, но время ее выполнения (около 30 с) значительно превышает $\Delta t = 1$ с, а при большем шаге возникает неустойчивость. Поэтому для полетов тяжелых самолетов можно пренебречь моделированием действия рулей высоты (рис. 47) и воспользоваться следующими расчетными формулами:

$$x_{i+1} = x_i + \Delta x_i - B \Delta x_i |\Delta x_i| + P_i;$$

$$h_{i+1} = h_i + (\Delta h_i + q C e^{A h_i} (\Delta x_i)^2 - G) \Delta h_{\max} / (q C e^{A h_i} (\Delta x_i)^2),$$

клавиши БП 4 9 С/П; для последующих пусков — клавиши В/О и С/П или С/П; после каждого выполнения программы (время счета около 15 с) $PX = V_x$, $PY = P2 = h_i$; $P3 = \Delta h_i$; $P4 = i$; $P5 = x_i$; $P6 = \Delta x_i$ ($k = 0,36$ для V_x , км/ч; $q_0 = 1$).

где $G = 9,81 (\Delta t)^2$.

Эти формулы реализованы программой 60, при составлении которой предусмотрено использование «автопилота» для вычисления координат через $n > 1$ шагов Δt , а структура формулы для приближенного вычисления h_{i+1} обеспечивает устойчивость вычислений при большом шаге $\Delta t \geq 10$ с.

При использовании микро-ЭВМ с большими быстродействием и ресурсом памяти можно существенно повысить точность моделирования инерционных объектов. Эффективность такого моделирования повышается при использовании персональных ЭВМ, допускающих отображение на дисплее текущей траектории движения управляемого объекта.

СПИСОК ЛИТЕРАТУРЫ

1. Гарднер М. Математические головоломки и развлечения.— М.: Мир, 1971.— 510 с.
2. Гарднер М. Математические досуги.— М.: Мир, 1971.— 496 с.
3. Гарднер М. Математические новеллы.— М.: Мир, 1974.— 454 с.
4. Гик Е. Я. Шахматы и математика.— М.: Наука, 1983.— 174 с.
5. Доморяд А. П. Математические игры и развлечения.— М.: Физматгиз, 1961.— 266 с.
6. Кетков Ю. Я. Программирование на БЕИСИКЕ.— М.: Статистика, 1978.— 156 с.
7. Романовский Т. Б. Микрокалькуляторы в рассказах и играх.— Рига: Зинатне, 1984.— 120 с.
8. Трохименко Я. К., Любич Ф. Д. Инженерные расчеты на микрокалькуляторах.— К.: Техника, 1980.— 384 с.
9. Трохименко Я. К., Любич Ф. Д. Инженерные расчеты на программируемых микрокалькуляторах.— К.: Техника, 1985.— 328 с.
10. Трохименко Я. К., Любич Ф. Д. Микрокалькулятор, Ваш ход! — М.: Радио и связь, 1985.— 216 с.
11. Трохименко Я. К., Любич Ф. Д. Радиотехнические расчеты на микрокалькуляторах.— М.: Радио и связь, 1983.— 256 с.
12. Цветков А. Н., Епанечников В. А. Прикладные программы для микро-ЭВМ «Электроника Б3-34», «Электроника МК-56», «Электроника МК-54».— М.: Финансы и статистика, 1984.— 174 с.
13. Комски Д. М. Хори, игры и автомати.— София: Техника, 1974.— 174 с.

ОГЛАВЛЕНИЕ

Предисловие	3
Глава 1 ИГРЫ И КИБЕРНЕТИКА	5
Глава 2 ВХОДНЫЕ ЯЗЫКИ МИКРО-ЭВМ	15
Глава 3 РЕШЕНИЕ ЛОГИЧЕСКИХ ЗАДАЧ	25
Глава 4 ИГРЫ С ПОЛНОЙ ИНФОРМАЦИЕЙ	43
Глава 5 ИГРЫ С НЕПОЛНОЙ ИНФОРМАЦИЕЙ	66
Глава 6 МИКРО-ЭВМ — СУДЬЯ	80
Глава 7 ИГРЫ С МОДЕЛИРОВАНИЕМ ДВИЖЕНИЯ ИНЕРЦИОННЫХ ОБЪЕКТОВ	99
Список литературы	119

Научно-популярное издание

ЯРОСЛАВ КАРПОВИЧ ТРОХИМЕНКО
д-р техн. наук

ИГРЫ С МИКРО-ЭВМ

Редактор О. П. Веремейчик
Оформление художника А. А. Белянина
Художественный редактор
В. С. Шапошников
Технические редакторы С. В. Иванус,
Н. А. Бондарчук
Корректоры Г. Г. Бондарчук,
Н. Г. Петрик

Информ. бланк № 3316

Сдано в набор 19.03.86. Подписано в печать 12.09.86.

БФ 05857. Формат 70 × 90/16. Бумага офс. № 1. Гарн.

школьная. Печ. офс. Усл. печ. л. 8,77. Усл. кр.-отт. 32,19.

Уч.-изд. л. 9,63. Тираж 35 000 экз. Зак. 6—1166. Цена 55 к.

Издательство «Техника», 252601, Киев, 1, Крещатик, 5.

Головное предприятие республиканского производственного

объединения «Полиграфкнига», 252057, Киев-57, ул. Дов-

женко, 3.

Киев
«Техніка»
1986

