**СОДЕРЖАНИЕ**

[СИСТЕМНОЕ ПРОГРАММИРОВАНИЕ 2](#_Toc30175498)

[Системное программирование средствами Python 2](#_Toc30175499)

[ПРИКЛАДНОЕ ПРОГРАММИРОВАНИЕ 2](#_Toc30175500)

[Сравнительный анализ интегрированных сред разработки: NetBeans, Eclipse, IntelliJ Idea. 2](#_Toc30175501)

[Классы в Java 6](#_Toc30175502)

[Метод PSVM 7](#_Toc30175503)

[Комментарии в Java 8](#_Toc30175504)

[объявление переменных в java 8](#_Toc30175505)

[java ввод строки через консоль 8](#_Toc30175506)

[Трассировка проэкта 8](#_Toc30175507)

# СИСТЕМНОЕ ПРОГРАММИРОВАНИЕ

## Системное программирование средствами Python

Встроенные в Python интерфейсы доступа к службам операционных систем делают его идеальным инструментом для создания переносимых программ и утилит системного администрирования. Программы на языке Python могут отыскивать файлы и каталоги, запускать другие программы, производить параллельные вычисления с использованием нескольких процессов и потоков и делать многое другое.

Стандартная библиотека Python полностью отвечает требованиям стандартов POSIX и поддерживает все типичные инструменты операционных систем: переменные окружения, файлы, сокеты, каналы, процессы, многопоточную модель выполнения, поиск по шаблону с использованием регулярных выражений, аргументы командной строки, стандартные интерфейсы доступа к потокам данных, запуск команд оболочки, дополнение имен файлов и многое другое.

Кроме того, системные интерфейсы в языке Python созданы переносимыми, например сценарий копирования дерева каталогов не требует внесения изменений, в какой бы операционной системе он ни использовался. Система Stackless Python, используемая компанией EVE Online, также предлагает улучшенные решения, применяемые для параллельной обработки данных.

# ПРИКЛАДНОЕ ПРОГРАММИРОВАНИЕ

## СРАВНИТЕЛЬНЫЙ АНАЛИЗ ИНТЕГРИРОВАННЫХ СРЕД РАЗРАБОТКИ: NETBEANS, ECLIPSE, INTELLIJ IDEA.

### NetBeans

NetBeans появилась как студенческий университетский проект в Праге в 1996 году. В 1997 году IDE стала коммерческим продуктом, а в 1999 году её выкупила компания Sun Microsystems (родители Java) и уже на следующий год представила open source-релиз.

Актуальная версия 8.1 работает на машинах под управлением ОС Windows, Mac OS X, Linux и Solaris. Ну а пакет portable можно запустить на любых системах, для которых существует Java-машина. Себе я загрузил Java EE bundle, это один из шести возможных пакетов загрузки. Этот бандл поддерживает JavaScript и HTML, GlassFish и Tomcat, но не поддерживает PHP, C/C++/Fortran, Groovy и Grails: их можно получить в пакете «Всё включено» (или просто «All»). Тем не менее, при желании, я в любой момент смогу загрузить поддержку вышеназванных языков, выбрав соответствующий плагин (да и любой другой). Их у NetBeans поменьше, чем у Eclipse, зато они обычно не конфликтуют друг с другом.

Этой осенью Oracle (ей NetBeans досталась после поглощения Sun Microsystems) решила передать эту среду разработки под крыло Apache Software Foundation вместе со всеми правами, исходными кодами, торговой маркой, доменом "netbeans.org" и рядом других элементов инфраструктуры. Будущее проекта пока туманно, хотя раньше у системы были определенные родственные привилегии. Так, именно NetBeans первой получила поддержку Java 8 практически сразу после выхода обновленной платформы, и была названа «официальной IDE для Java 8». Впрочем, через месяц после выхода это преимущество было утеряно: именно тогда другие IDE также получили поддержку восьмой Java.

Тем не менее, поддержка Java 8 в NetBeans действительно хороша, и эта IDE отлично подходит для вплетения в «старый» код трюков восьмой версии. Её редакторы, анализаторы кода и конвертеры помогут программисту провести апгрейд кода, используя в нем конструкции, характерные для Java 8 — лямбда-выражения, функциональные операторы и ссылки на методы. Плагины JavaScript в NetBeans 8 отлично справляются с поддержкой Node.js и новейших инструментов JavaScript, таких как Gulp и Mocha, равно как и поддержку интерпретатора JavaScript Nashorn.

Редактор NetBeans поддерживает языки, обнаруживает ошибки в то время, когда вы печатаете, и помогает вам с помощью всплывающих подсказок и «умным» автодополнением кода. По субъективному ощущению IDE справляется с этой задачей быстрее, чем Eclipse, но несколько медлительнее IntelliJ IDEA. Кроме того, NetBeans обладает полным спектром инструментов рефакторинга (что показано на рисунке 3), которые позволяют программисту реструктуризировать код, не ломая его, выполнять анализ исходников, а также предлагает широкий набор подсказок для быстрых исправлений или расширения кода. В состав NetBeans входит инструмент проектирования для графического интерфейса пользователя Swing, ранее известный как "Project Matisse".

Разработчики высоко оценивают средство автоматизированного рефакторинга Inspect & Transform, появившееся в версии NetBeans 7.1. Оно позволяет провести анализ кода проекта и сделать предлагаемые улучшения. Хотя лично я предпочитаю сначала проверить весь собственный код unit-тестами, и только затем запускать инструменты, которые могут внести радикальные изменения. Я неоднократно страдал от всяческих автоматических исправлений, которые привели к невосполнимым последствиям.

### Eclipse IDE

Лет 10 назад на вопрос о лучшей IDE, Java-разработчик отвечал уверенно: Eclipse. Долгие годы эта среда разработки уверенно держала пальму первенства среди Java IDE. Эта среда полностью бесплатная, с открытым исходным кодом, написанным преимущественно на Java. Тем не менее, её модульная архитектура позволяет использовать Eclipse и с другими языками. Проект Eclipse, инициированный IBM, появился в 2001 году. Им хотели заменить семейство сред разработки IBM Visual Age, основанных на Smalltalk.

Ну а главной целью, о чем даже название говорит, было затмить Microsoft Visual Studio (eclipse по-английски означает затмение).

Портативность Java помогает Eclipse быть кроссплатформенной средой: эта IDE работает на Linux, Mac OS X, Solaris и Windows.

Хорошо это или плохо, Java Standard Widget Toolkit (SWT), по крайней мере частично, отвечает за внешний вид Eclipse.

Своей производительностью (или, как говорят некоторые доброжелатели, её отсутствию) Eclipse обязана JVM. Eclipse работает довольно медленно, поскольку упирается корнями в довольно старое «железо» и древние версии JVM. Даже сегодня она кажется медлительной, особенно если нацепить на неё много плагинов.

Часть расходов ресурсов Eclipse можно отнести на счёт её встроенного инкрементного компилятора, который запускается всякий раз при загрузке файла или обновлении кода. Полезная штука, именно она ловит ошибки при вводе текста.

Независимо от сборки, проект Eclipse поддерживает модель контента, которая содержит информацию об иерархии типов, ссылок и объявлениях Java-элементов.

Текущая версия Eclipse содержит минимальную Eclipse SDK, а плагины добавляются по требованию. К слову, работа с плагинами в этой IDE — не для слабонервных. Сторонние плагины часто конфликтуют между собой, хотя в их официальной спецификации об этом ничего не сказано.

Экосистема плагинов Eclipse — это одновременно сильная сторона этой IDE и одна из главных ее проблем. Именно из-за несовместимости плагинов порой падают целые сборки, и программистам приходится начинать работу сначала.

Первый опыт работы с Eclipse, может привести в замешательство, и даже сбить с толку. Поначалу необходимо настроить Eclipse и привыкнуть к её концептуальной архитектуре рабочих пространств, ракурсов и видов. Всё это определяется плагинами, которые вы установили. Для серверной разработки на Java, вы, вероятно, будете использовать ракурсы Java, Java EE и Java browsing, вид, отображающий структуру пакета (Package Explorer), ракурс отладки, ракурс командной синхронизации веб-инструментов, ракурс разработки баз данных и ракурс отладки базы данных. На практике все обретает смысл, когда вы откроете нужные вам окна.

Eclipse практически всегда предлагает несколько способов решения той или иной задачи. Например, вы можете просматривать код с помощью ракурса просмотра Java (Java browsing perspective). Что выбрать — дело вкуса и выбора.

Специальный поиск Java позволяет найти объявления, ссылки и вхождения Java-пакетов, типов, методов, полей. Вы также можете использовать быстрый доступ к поиску и предпросмотр.

Распространенные паттерны кода можно сгенерировать из шаблонов кода. Рефакторинг Java в Eclipse, поддерживает 23 операции, начиная от общепринятых операций по переименованию и заканчивая менее очевидными преобразованиями (как в книге Мартина Фаулера).

Eclipse, поддерживает отладку как локально, так и удаленно, при условии, что вы используете JVM, которая поддерживает удаленную отладку. Отладка довольно стандартна: вы определяете контрольные точки, а затем просматриваете переменные на вкладке отладки. Конечно, можно пошагово выполнять свой код и вычислять выражения.

У Eclipse — обширнейшая база документации самого разного возраста, ценности и полезности. Увы, обнаружить не соответствующую текущей версии картинку в инструкции, например, с устаревшим интерфейсом и расположением кнопок — обычное дело для этой IDE. К сожалению, проблема запоздалого обновления документации очень характерна для любых проектов с исходным кодом.

### IntelliJ IDEA

IntelliJ IDEA с точки зрения возможностей и цены поставляется в двух вариантах: бесплатного Community edition, и платного Ultimate edition с расширенной функциональностью.

Community edition предназначена для JVM- и Android-разработки. Бесплатная версия поддерживает Java, Kotlin, Groovy и Scala; Android; Maven, Gradle и SBT; работает с системами контроля версий Git, SVN, Mercurial и CVS.

Ultimate edition приспособлена для веб- и enterprise-разработки. Эта версия IDE работает не только с Git, SVN, Mercurial и CVS, но также с Perforce, ClearCase и TFS; в ней вы сможете писать на JavaScript и TypeScript; есть поддержка Java EE, Spring, GWT, Vaadin, Play, Grails и ряда других фреймворков. И, конечно, не обошлось без SQL и инструментов для работы с базами данных.

Идея, которой руководствуются разработчики этой IDE, формируя ценовую политику, заключается в том, что её коммерческая версия (Ultimate) займет свое место на компьютерах профессионалов, за счет чего их производительность повысится. Если Java-программист ежегодно зарабатывает $50 тысяч, возврат потраченных на платную IDE IntelliJ IDEA инвестиций (500 долларов за годовую подписку) произойдет очень быстро за счет даже незначительного ускорения его работы.

В последующие годы цена для бизнеса падает, для стартапов, фрилансеров она существенно ниже, а для студентов, учителей, Java-чемпионов и open source-разработчиков она и вовсе бесплатна.

IntelliJ IDEA подкупает своим глубоким пониманием кода, умной эргономикой, встроенными функциями для разработки и поддержкой многих языков.

Одной и важных особенностей является глубокое понимание кода. Подсветка синтаксиса и простое автодополнение кода — обычное дело для любых современных Java-редакторов. IDEA пошла дальше, предлагая «умное автодополнение». Этот термин означает, что среда разработки показывает список наиболее релевантных символов, применимых в данном контексте. Список символов зависит не только от контекста как такового, «общепринятого», но от стиля программирования разработчика, от того, насколько часто он использует те или иные операторы. «Завершение цепочки» и вовсе показывает список применимых символов, допустимых через методы или геттеры в текущем контексте. Кроме того, в случае со статическими членами или константами IDEA автоматически добавляет любые необходимые операторы импорта (import). Во всех случаях автодополнения, IDEA пытается угадать тип символа во время выполнения, уточнить свой выбор и даже применить приведение типов если необходимо.

Код Java часто включает фрагменты из других языков в виде строк. IDEA может вводить код SQL, XPath, HTML, CSS или JavaScript в строковые литералы Java. В этом смысле IDE может проводить рефакторинг кода на нескольких языках. Например, если вы переименуете класс в JPA-отображении, IDEA обновит соответствующий класс сущностей и выражений JPA.

Во время рефакторинга фрагмента кода, у разработчика возникает одно (вполне естественное) желание: чтобы все дубликаты этого кода также зарефакторились. IDEA Ultimate находит дубликаты и похожие фрагменты и также применяет к ним рефакторинг.

IntelliJ IDEA анализирует код при загрузке и непосредственно при вводе. Она указывает на предполагаемые проблемы (как на нашем рисунке выше) и, по желанию, предлагает список вероятных быстрых правок к обнаруженным проблемам.

Разработчики IDEA продумали и расширили поддержку кода Spring, Java EE, Grails, Play, Android, GWT, Vaadin, Thymeleaf, Android, React, AngularJS и других фреймворков. Вы, вероятно, заметили, что не все из них относятся к Java. IDEA непосредственно из коробки «понимает» и другие языки — Groovy, Kotlin, Scala, JavaScript, TypeScript и SQL. Если вы не нашли в этом списке нужного вам языка, в настоящий момент есть 19 языковых плагинов IntelliJ, в частности, для поддержки R, Elm и D.

После тестирования именно от IntelliJ IDEA Ultimate у возникает ощущение правильной среды разработки. IDEA — лучшая IDE для Java. Хоть она и не бесплатна, прирост производительности от ее использования стоит годовой подписки.

Для начинающих, которые не могут себе позволить годовую подписку IntelliJ IDEA Ultimate, рекомендуется использовать NetBeans, а не Eclipse. Конечно, экосистема плагинов Eclipse сегодня развита куда больше, чем у любой другой IDE однако она очень заросшая и неухоженная: начинающий разработчик рискует увязнуть в дебрях IDE, Java и погружение в работу уйдут на второй план.

## КЛАССЫ В JAVA

Java является объектно-ориентированным языком, поэтому такие понятия как "класс" и "объект" играют в нем ключевую роль. Любую программу на Java можно представить как набор взаимодействующих между собой объектов.

Шаблоном или описанием объекта является класс, а объект представляет экземпляр этого класса. Можно еще провести следующую аналогию. У нас у всех есть некоторое представление о человеке - наличие двух рук, двух ног, головы, туловища и т.д. Есть некоторый шаблон - этот шаблон можно назвать классом. Реально же существующий человек (фактически экземпляр данного класса) является объектом этого класса.

Класс определяется с помощью ключевого слова class.

Кроме обычных методов классы могут определять специальные методы, которые называются конструкторами. Конструкторы вызываются при создании нового объекта данного класса. Конструкторы выполняют инициализацию объекта.

Если в классе не определено ни одного конструктора, то для этого класса автоматически создается конструктор без параметров.

Если конструктор не инициализирует значения переменных объекта, то они получают значения по умолчанию. Для переменных числовых типов это число 0, а для типа string и классов - это значение null.

Если необходимо, что при создании объекта производилась какая-то логика, например, чтобы поля класса получали какие-то определенные значения, то можно определить в классе свои конструкторы.

Ключевое слово this представляет ссылку на текущий экземпляр класса. Через это ключевое слово можно обращаться к переменным, методам объекта, а также вызывать его конструкторы.

Кроме конструктора начальную инициализацию объекта вполне можно было проводить с помощью инициализатора объекта. Инициализатор выполняется до любого конструктора. То есть в инициализатор можно поместить код, общий для всех конструкторов.

## МЕТОД PSVM

Метод PSVM запускает программу.

public static void main (String[] args){...}

1. public - доступен для всех из любого класса и пакета
2. static - это идентификатор означающий что наш метод ну или поле в единственном экземпляре , т.е если мы создадим идентичный метод или переменную то при вызове будет использована наша первая написанная статическая переменная или статичный метод
3. void - означает что он принимает данные и что то с ними делает
4. main - имя метода
5. String[] args - это массив аргументов с которыми может запускаться программа. По умолчанию массив пуст.

## ОФОРМЛЕНИЕ КОММЕНТАРИЕВ В JAVA

Комментарии - казалось бы, что может быть проще. И чего тут целую статью писать. Но тут не всё так просто. Как говорил мой начальник, код писать может каждый, а вот хороший комментарий написать это сложно.

Комментарии в Java: не всё так просто - 1

Большинство курсов по изучению языка начинаются с традиционного Hello World. Даже в Oracle Tutorials в разделе «Getting Started» мы начинаем с The "Hello World!" Application. И с самых первых строк кода мы видим их – Java комментарии.

Их важность так же подчёркивается тем, что в таком важном документе, как «Java Code Convention» комментариям отводится отдельный раздел: Comments.

Как гласит документация, комментарии в Java делятся на два типа:

комментарий реализации (или комментарий кода);

документирующий комментарий.

Комментарии кода используются для описания отдельных строк/блоков, а комментарии для документирования используются, чтобы описать спецификацию кода (его интерфейс), не зависящую от его реализации.

Java комментарии игнорируются компилятором, т.к. они несут смысл для разработчика, а не для пользователя. Поэтому, можно уменьшить размер компилируемых классов.

Из названия понятно, что данный комментарий относится к коду и должен отражать его особенности.

Комментарии кода бывают:

1. Строчные (т.е. описываются в одну строку)

// Строчный комментарий

1. Блочные (т.е. описываются целым блоком, т.к. не помещаются в одну строку)

/\*

\* Блочный комментарий

\*/

Интересной особенностью блочного комментария является то, что если мы начнём его с «/\*-» (т.е. добавим после астериска минус), то текст данного блочного комментария отформатирован не будет.

Интересно, но при помощи определённых комментариев можно дать некоторым IDE определённые подсказки. Например, при помощи строчных комментариев «//@formatter:on» и «//@formatter:off» в IDE Eclipse можно отключить форматирование для участков кода.

## ОБЪЯВЛЕНИЕ ПЕРЕМЕННЫХ В JAVA

В мире магии во-первых, первично сознание, а не материя. А во-вторых, может существовать любая мыслимая сущность. Поэтому сотворение сущности начинается с утверждения, что она существует. Это называется – объявлением. А так как делать маг с сотворенной сущностью может что угодно и когда угодно, в том числе менять ее значение, то называются объявляемые сущности переменными.

Так как даже начинающий маг много чего себе может напридумывать, сущности различаются и систематизируются с помощью типов. Есть типы созданные Создателем, их изменить нельзя и они называются примитивными. Все остальные типы создает сам маг и они называются объектными.

## ВВОД СТРОКИ ЧЕРЕЗ КОНСОЛЬ

Для получения ввода с консоли в классе System определен объект in. Однако непосредственно через объект System.in не очень удобно работать, поэтому, как правило, используют класс Scanner, который, в свою очередь использует System.in. Например, напишем маленькую программу, которая осуществляет ввод чисел:

Scanner in = new Scanner(System.in);

System.out.print("Input a number: ");

int num = in.nextInt();

System.out.printf("Your number: %d \n", num);

in.close();

## ТРАССИРОВКА ПРОЭКТА

1. Запуск отладчика

После того как вы настроите конфигурацию запуска вашего проекта, вы можете запускать его в режиме отладки, нажав Shift + F9

Руководство пользователя IntelliJ IDEA. Отладчик - 2

В окне отладчика вы можете видеть стек вызовов функций и список потоков, с их состояниями, переменными и окнами просмотра состояния. Когда вы выбираете контекст вызова функции, вы можете просмотреть значения переменных соответствующих выбранному контексту.

1. Полезные клавиатурные сокращения отладчика

Установить/снять точку останова - Ctrl + F8 (Cmd + F8 для Mac)

Возобновить выполнение программы - F9

Перейти к следующей инструкции - F8

Перейти внутрь функции - F7

Приостановить выполнение - Ctrl + F2 (Cmd + F2)

Переключить между просмотром списка точек останова и подробной информацией о выбранной точке - Shift + Ctrl + F8 (Shift + Ctrl + F8)

Запустить отладку кода с точки на которой стоит курсор - Shift + Ctrl + F9 (если это внутри метода main())

1. Умный переход внутрь

Иногда вам надо при пошаговой отладке перейти внутрь определенного метода, но не первого который будет вызван. В таком случае вы можете нажать Shift + F7 (Cmd + F7 для Mac) чтобы выбрать из предложенного списка метод который вам нужен. Это может сэкономить вам массу времени.

1. Удалить контекст вызова функции

Если вам нужно «вернуться назад во времени» во время отладки, вы можете сделать это удалив контекст вызова функции. Это сильно поможет если вы по ошибке зашли слишком глубоко. Таким образом вы не откатите глобальное состояние выполнения программы, но как минимум вы вернетесь назад по стеку вызовов функций.

Руководство пользователя IntelliJ IDEA. Отладчик - 4

1. Переход к курсору

Иногда вам надо возобновить выполнение программы и остановиться на какой-то другой строчке кода, не создавая точку останова. Это легко - просто нажмите Alt + F9.

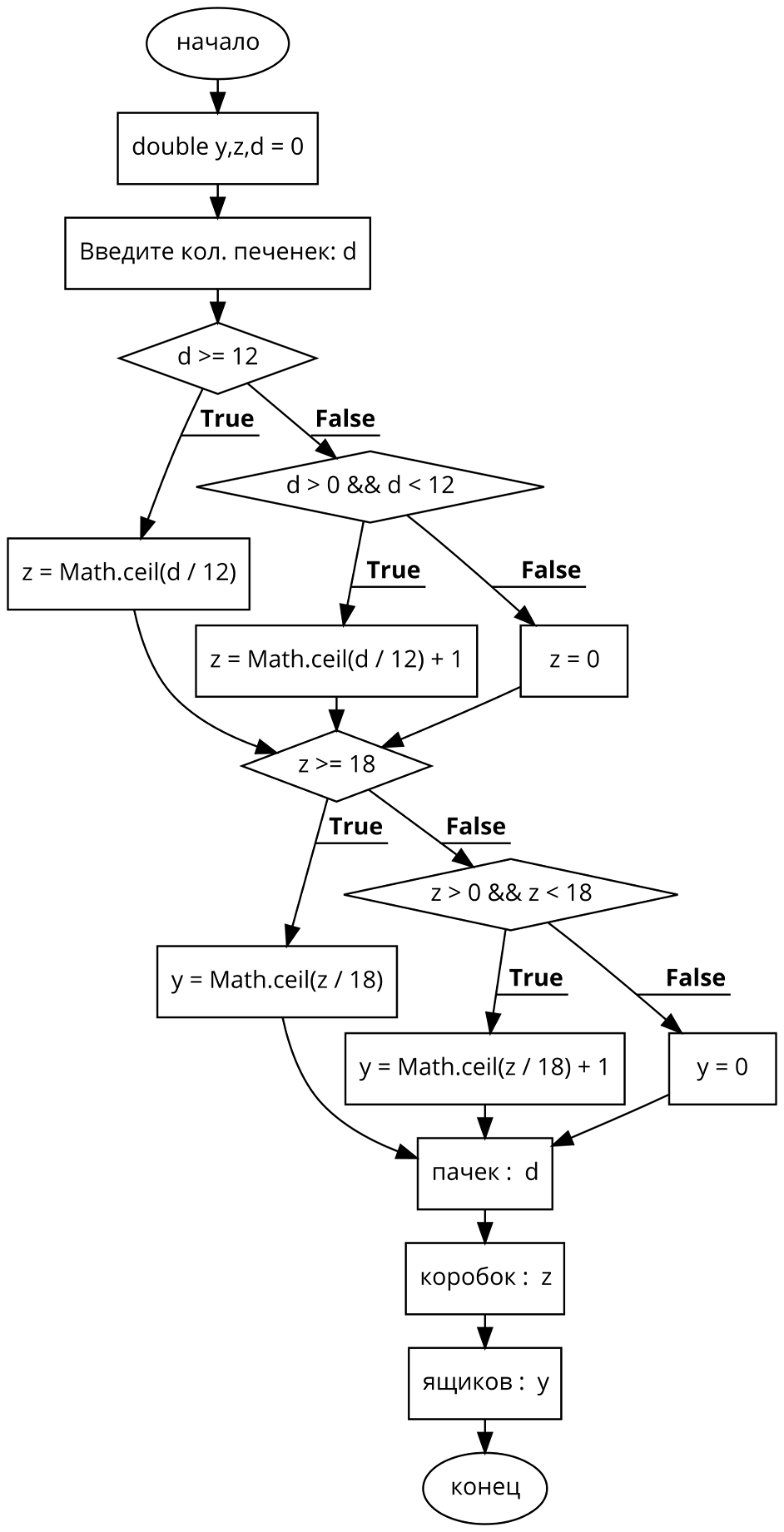
# **СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ**

1. Что можно реализовать с помощью Python? [Электронный ресурс] – Режим доступа: http://python-3.ru/page/chto-mozhno-delat-s-pomoshhju-python#cut(17.01.2020)
2. Eclipse, NetBeans или IntelliJ IDEA? Выбираем IDE для Java-разработки [Электронный ресурс] – Режим доступа: https://javarush.ru/groups/posts/1642-eclipse-netbeans-ili-intellij-idea-vihbiraem-ide-dlja-java-razrabotki(17.01.2020)
3. Java | Классы и объекты [Электронный ресурс] – Режим доступа: https://metanit.com/java/tutorial/3.1.php(17.01.2020)
4. Создание и запуск первого Java-приложения (часть 2) [Электронный ресурс] – Режим доступа: https://javarush.ru/groups/posts/694-sozdanie-i-zapusk-pervogo-java-prilozhenija-chastjh-2(17.01.2020)
5. Комментарии в Java : как их писать [Электронный ресурс] – Режим доступа: https://javarush.ru/groups/posts/607-kommentarii-v-java--ne-vsje-tak-prosto(17.01.2020)
6. Руководство пользователя IntelliJ IDEA. Отладчик [Электронный ресурс] – Режим доступа: https://javarush.ru/groups/posts/753-rukovodstvo-poljhzovatelja-intellij-idea-otladchik(17.01.2020)

# ПРИЛОЖЕНИЕ А

A

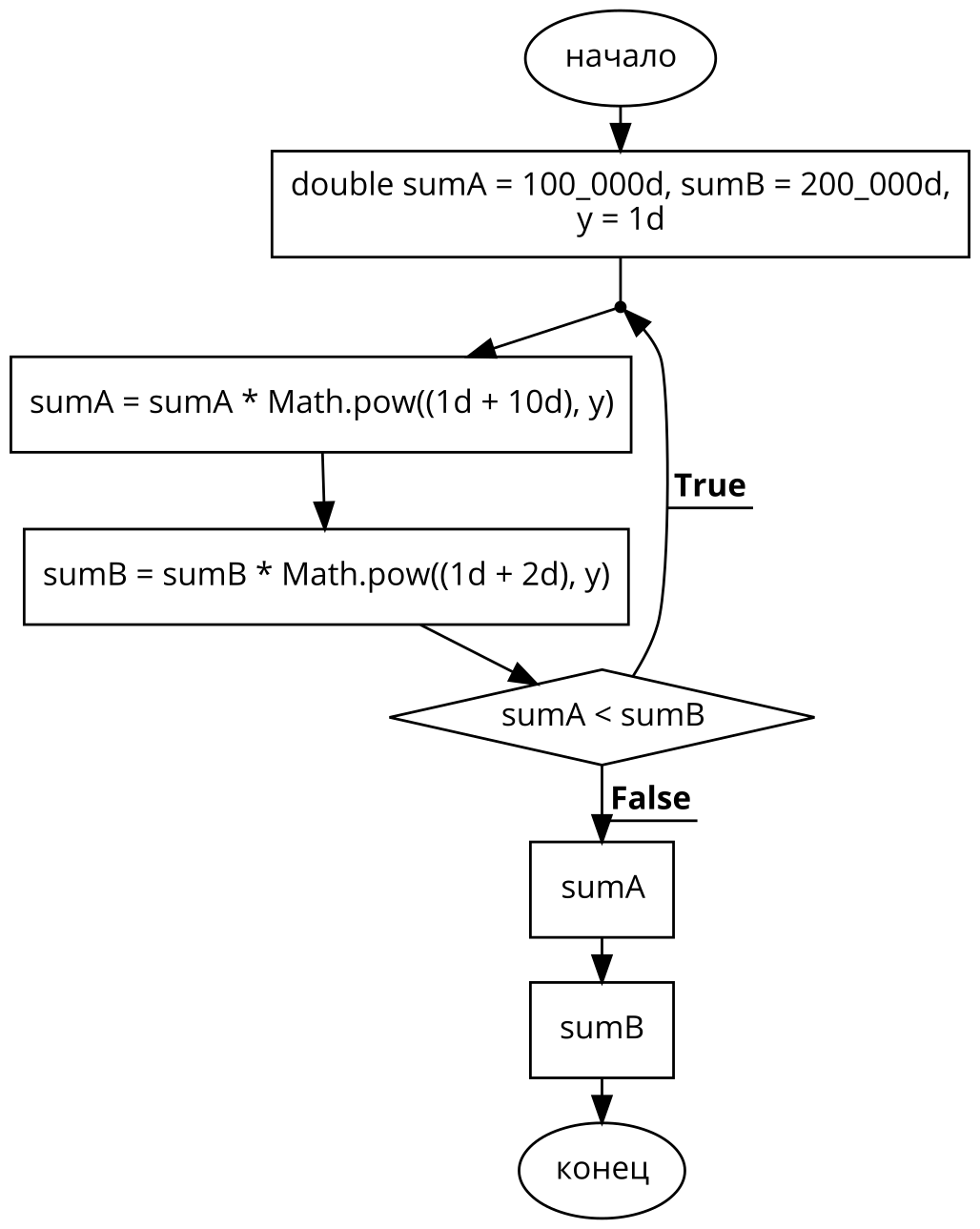
1. Объявление переменных
2. Расчёт кол. Пачек
3. Расчёт кол. Коробок
4. Расчёт кол. Ящиков
5. Вывод инф.



B

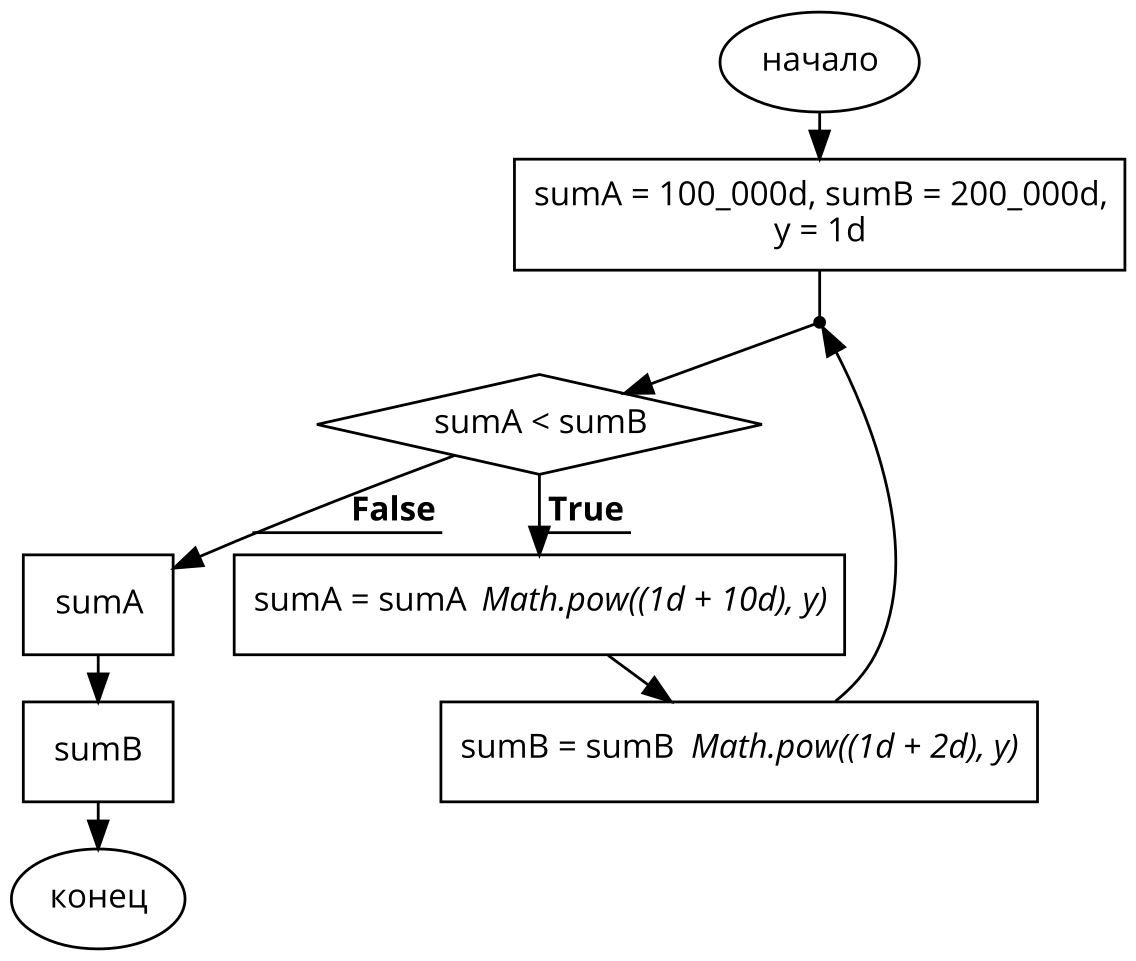
DO WHILE

1. Объявление переменных
2. Расчёт вкладов пока 1 не превысит 2
3. Вывод 2 вкладов



WHILE

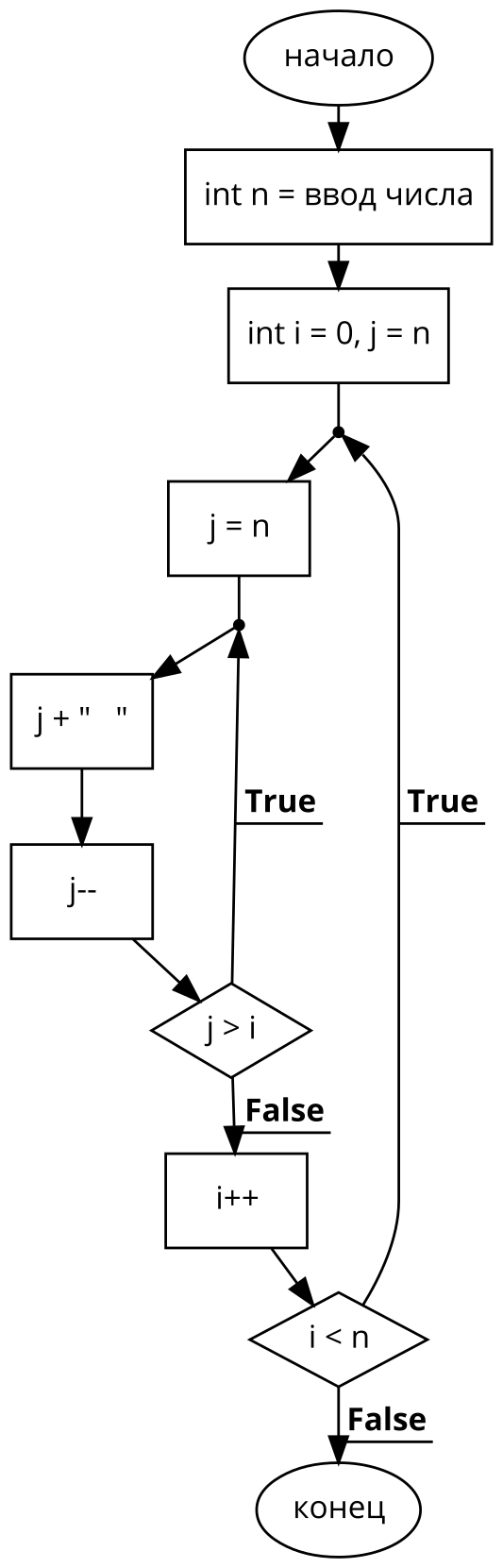
1. Объявление переменных
2. Расчёт вкладов пока 1 не превысит 2
3. Вывод 2 вкладов



C

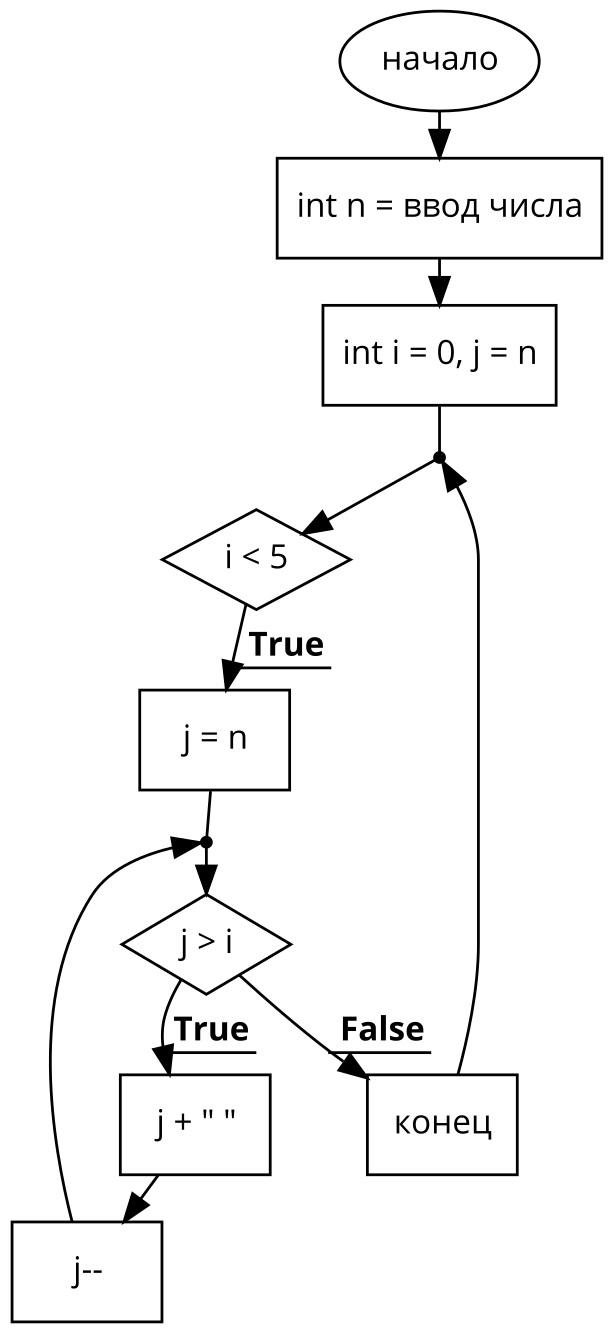
DO WHILE

1. Вывод чисел от числа ввода до 1 с удалением последнего числа



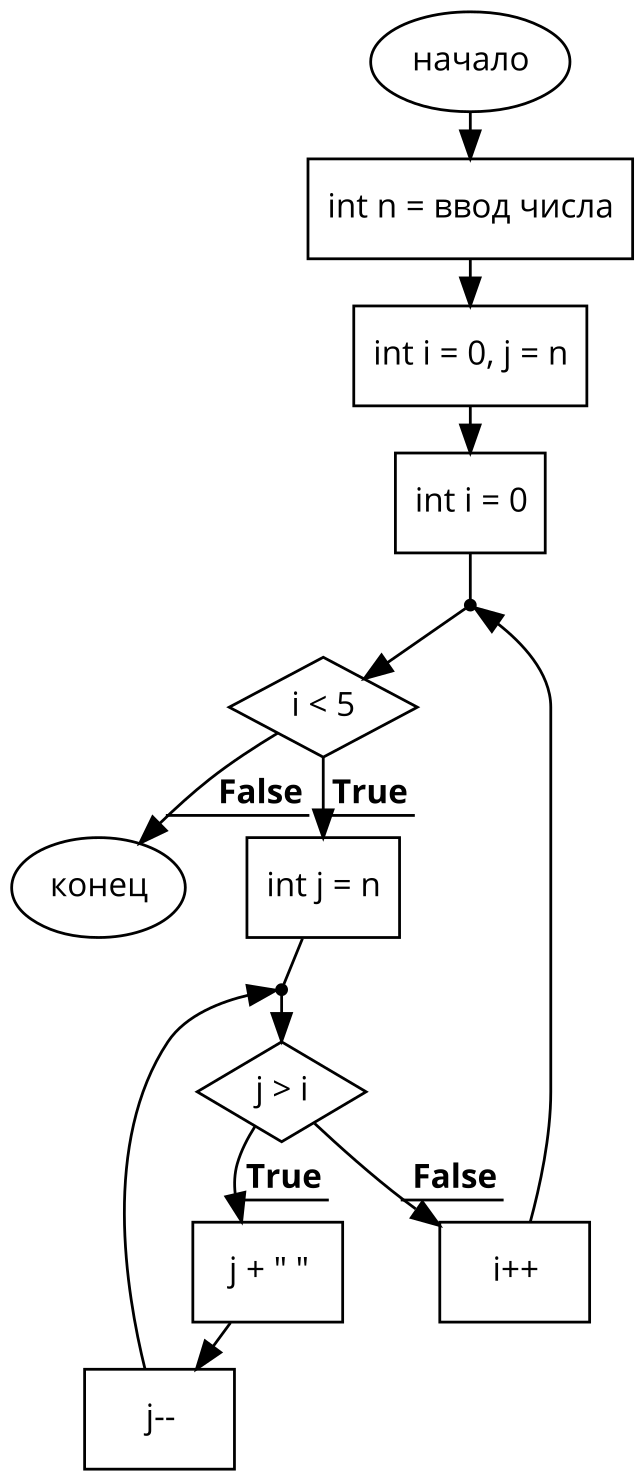
WHILE

1. Вывод чисел от числа ввода до 1 с удалением последнего числа



FOR

1. Вывод чисел от числа ввода до 1 с удалением последнего числа



# ПРИЛОЖЕНИЕ Б