МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ЮЖНО-УРАЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ГУМАНИТАРНО-ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ»

(ФГБОУ ВО «ЮУрГГПУ»)

ФИЗИКО-МАТЕМАТИЧЕСКИЙ ФАКУЛЬТЕТ

КАФЕДРА ИНФОРМАТИКИ, ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И МЕТОДИКИ ОБУЧЕНИЯ ИНФОРМАТИКЕ

Методические особенности изучения современных языков программирования в школе

Выпускная квалификационная работа

по направлению 44.03.01 Педагогическое образование

Направленность программы бакалавриата

«Информатика»

Проверка на объем заимствований:
66,9 % авторского текста

Работа рекомендована к защите
рекомендована/не рекомендована
« 28 » мая 2019 г.
зав. кафедрой ИИТиМОИ
_____ Рузаков А.А.

Выполнила:
Студентка группы ЗФ-513-092-5-1
Жовтун Алёна Алексеевна

Научный руководитель:
к.п.н, доцент кафедры ИИТиМОИ
Носова Людмила Сергеевна

Челябинск
2019

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ЮЖНО-УРАЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ГУМАНИТАРНО-ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ»

(ФГБОУ ВО «ЮУрГГПУ»)

ФИЗИКО-МАТЕМАТИЧЕСКИЙ ФАКУЛЬТЕТ

КАФЕДРА ИНФОРМАТИКИ, ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И МЕТОДИКИ ОБУЧЕНИЯ ИНФОРМАТИКЕ

Методические особенности изучения современных языков программирования в школе

Выпускная квалификационная работа

по направлению 44.03.01 Педагогическое образование

Направленность программы бакалавриата

«Информатика»

Выполнила:
Студентка группы ЗФ-513-092-5-1
Жовтун Алёна Алексеевна

Научный руководитель:
к.п.н, доцент кафедры ИИТиМОИ
_____
Носова Людмила Сергеевна

Проверка на объем заимствований:
_____% авторского текста

Работа _____ к защите
рекомендована/не рекомендована
« ___ » _____ 20__ г.
зав. кафедрой ИИТиМОИ

_____ Рузаков А.А.

Челябинск
2019

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Язык программирования для изучения в школе должен быть простой, понятный и иметь удобочитаемые конструкции. Излишняя гибкость, вольность синтаксиса затрудняют понимание кода программ учащимися. Поэтому в образовательном процессе предпочитают языки семейства Pascal как достаточно простые в изучении.

В данной дипломной работе, мы будем рассматривать наиболее популярные современные языки программирования R, Python, C#, C++ и Java. С данными языками ученики впервые знакомятся в школе. Также рассмотрим основные критерии выбора языка программирования и его образовательные возможности.

При выборе языка программирования в образовательных целях не играют роли: его новизна, эффективность реализации (в виде компилятора или интерпретатора), специфические возможности и популярность.

В связи с наблюдаемым на текущий момент быстрым развитием персональной вычислительной техники, происходит постепенное изменение требований, предъявляемых к языкам программирования. В связи с возрастающей мощностью современных компьютеров, программы на интерпретируемых языках остаются достаточно простыми, а скорость работы догоняет скомпилированные программы.

Явный интерес представляет к рассмотрению язык программирования Python (пайтон). Основным разработчиком языка Python является Гвидо ван Россум (Guido van Rossum), а первая стабильная версия 1.0 появилась в январе 1994 года [1].

Цель дипломной работы: рассмотреть методические особенности изучения современных языков программирования в школе и разработать курс «Основы программирования на Python» и программно-поддержку к нему.

Объект исследования: язык программирования Python.

Предмет исследования: процесс изучения языка программирования Python в старшей школе.

Задачи дипломной работы:

1. Провести анализ понятия языка программирования.
2. Рассмотреть особенности современных языков программирования, в т.ч. языка программирования Python.
3. Провести анализ языков программирования, изучаемые в школе; сделать выводы по изученной информации.
4. Провести анализ нормативных документов и разработать курс по выбору «Основы программирования на Python».
5. Разработать программно-методическую поддержку курса в виде электронного пособия.

Гипотеза: если в программу среднего общего образования, разработать и включить курс «Основы программирования на Python», то это будет способствовать овладению учащимися основами логического и алгоритмического мышления, формировать базовые понятия структурного программирования, развивать аналитический стиль мышления начинающих программистов, а также умение самостоятельно составлять алгоритмы решения задач и формировать правильный стиль мышления.

ГЛАВА 1. ТЕОРЕТИЧЕСКИЕ ОСНОВЫ СОВРЕМЕННЫХ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ

1.1. Понятие языка программирования

Язык программирования – это формальная знаковая система, предназначенная для написания программ для компьютера [10].

На данный момент существует больше двух с половиной тысяч языков программирования. Каждый год это число возрастает. Существуют языки программирования, созданные разработчиками для собственных нужд, а другие становятся известны миллионам людей.

Обычный разговорный язык состоит из четырех основных элементов: символов, слов, словосочетаний и предложений. Язык программирования содержит подобные элементы, только слова называют элементарными конструкциями, словосочетания – выражениями, предложения – операторами.

Любой язык программирования состоит из набора правил – синтаксических, лексических и семантических:

- Алфавит – фиксированный для данного языка набор символов (букв, цифр, специальных знаков и т.д.), которые могут быть использованы при написании программы.
- Синтаксис – правила построения из символов алфавита специальных конструкций, с помощью которых составляется алгоритм.
- Семантика – система правил толкования конструкций языка.

Любая программа для компьютера составляется с помощью символов алфавита с помощью синтаксических и семантических правил.

Первые языки программирования были очень примитивными, а код программ сильно напоминал последовательность нолей и единиц. Программистам было нелегко писать программы, так как они должны были знать числовые коды всех машинных команд и самостоятельно распределять память под команды в программе и данные. Позже, чтобы облегчить общение человека с ЭВМ, создан язык программирования

Assembler. Появились словесные обозначения ссылок на ячейки памяти, содержащие данные, мнемонические обозначения заменили числовые коды операций, которые легче запомнить. Программирование приблизилось к написанию кода, как будто на человеческом языке.

Первый язык программирования высокого уровня – Fortran (Formula Translation) был создан в середине 50-х годов. Благодаря своей простоте и тому, что на этом языке накоплены большие библиотеки программ, Fortran используется для инженерных и научных расчетов, для решения задач физики и других наук с развитым математическим аппаратом.

Для решения экономических задач был создан язык программирования – Cobol.

Расширение областей применения ЭВМ влечет за собой создание новых языков, ориентированных на специфические области применения, такие как Snobol, Lisp. В исследованиях по созданию искусственного интеллекта, удобно использовать Lisp.

В 1968 г. объявили конкурс на лучший язык программирования для обучения студентов. Победителем стал язык Algol-68, но он не получил широкого распространения. Также для этого конкурса Никлаус Вирт создал достаточно удобный и простой, с наличием мощных средств структурирования данных язык Pascal. Для обучения младших школьников Самуэлем Пайпертом был разработан простой язык Logo.

Язык Basic получил широкое распространение в школах. Спустя много лет после изобретения, Basic и сегодня самый простой для освоения из десятков языков программирования.

В начале 70-х годов необходимость разработки больших, сложных и системных программ, потребовала создания специального языка программирования. Был разработан язык С. Он является одним из универсальных языков программирования. В отличие от Pascal, он давал больше контроля при работе с памятью и машинных команд. На С разрабатывались целые операционные системы, трансляторы, базы

данных, системные и прикладные программы. Программы, написанные на C, сравнимы по скорости с программами, написанными на языке Assembler, а код программ легче понимается.

Появление функционального программирования привело к созданию языка Prolog, разработанный для задач анализа и понимания естественных языков.

В 80-х годах был создан язык Ada. В дополнение к классическим свойствам языков тех времён, Ada решал задачи реального времени и имел инструменты для реализации параллелизма.

Деление языков программирования на низкого, высокого и сверхвысокого уровня является наиболее распространенной классификацией.

- Низкий уровень – языки Автокод и Assembler. Они являются машинно-ориентированными языками, т. к. команды соответствуют кодам исполняемым непосредственно на процессоре.
- Высокий уровень – Fortran, Algol, Cnobol, Pascal, Basic, C, Prolog и т.д. Эти языки машинно-независимы, т. к. переводом кода в машинные команды занимается компилятор и транслятор. Это вносит небольшое замедление в скорости по сравнению с языками низкого уровня.
- Сверхвысокий уровень – Algol-68 и APL. Операторы этих языков описывают гораздо больше действий по сравнению с языками высокого уровня.

Языки программирования также делятся на вычислительные (Fortran, Pascal, Algol, Basic, C) и языки символьной обработки(Лисп, Prolog, Cnobol)

Существует два основных направления развития языков программирования: процедурное и непроцедурное (декларативное).

Процедурное программирование возникло на заре вычислительной техники и получило широкое распространение. В процедурных языках

программа явно описывает действия, которые необходимо выполнить, а результат задается только способом получения его при помощи некоторой процедуры, которая представляет собой определенную последовательность действий.

Среди процедурных языков выделяют в свою очередь структурные и операционные языки. В структурных языках одним оператором записываются целые алгоритмические структуры: ветвления, циклы и т.д. В операционных языках для этого используются несколько операций.

Широко распространены следующие структурные языки: Pascal, C, Ada, Pl/1. Среди операционных известны Fortran, Basic, Focal.

Непроцедурное программирование появилось в начале 70-х годов 20-го века, но стремительное его развитие началось в 80-е годы, когда был разработан японский проект создания ЭВМ пятого поколения, целью которого явилась подготовка почвы для создания интеллектуальных машин. К непроцедурному программированию относятся функциональные и логические языки.

В функциональных языках программа описывает вычисление некоторой функции. Обычно эта функция задается как композиция других, более простых, те в свою очередь разлагаются на еще более простые и т.д. Один из основных элементов в функциональных языках – рекурсия, то есть вычисление значения функции через значение этой же функции от других элементов. Присваивания и циклов в классических функциональных языках нет.

В логических языках программа вообще не описывает действий. Она задает данные и соотношения между ними. После этого системе можно задавать вопросы. Машина перебирает известные и заданные в программе данные и находит ответ на вопрос. Порядок перебора не описывается в программе, а неявно задается самим языком. Классическим языком логического программирования считается Пролог. Построение логической программы вообще не требует алгоритмического мышления, программа

описывает статические отношения объектов, а динамика находится в механизме перебора и скрыта от программиста.

Можно выделить еще один класс языков программирования – объектно-ориентированные языки высокого уровня. На таких языках не описывают подробной последовательности действий для решения задачи, хотя они содержат элементы процедурного программирования. Объектно-ориентированные языки, благодаря богатому пользовательскому интерфейсу, предлагают человеку решить задачу в удобной для него форме. Примером такого языка может служить язык программирования визуального общения Object Pascal, C++, Java.

Языки описания сценариев, такие как Perl, Python, Rexx, Tcl и языки оболочек UNIX, предполагают стиль программирования, весьма отличный от характерного для языков системного уровня. Они предназначаются не для написания приложения с нуля, а для комбинирования компонентов, набор которых создается заранее при помощи других языков. Развитие и рост популярности Internet также способствовали распространению языков описания сценариев. Так, для написания сценариев широко употребляется язык Perl, а среди разработчиков Web-страниц популярен JavaScript.

1.2. Современные языки программирования

Рассмотрим и сравним пять самых распространенных и востребованных языков программирования.

• Python

За данным языком будущее. Он простой для понимания и применения: Python постепенно входит в учебную программу, можно сказать, вытесняя Pascal и прочие «деревянные» языки. Потом, это нейронные сети: в случае, если разработчик ударяется в машинное обучение, то тут же обращает свое внимание на Python. Потому, что, данный язык программирования оброс необходимым числом библиотек,

ориентированных на нейронные сети (Ruby в этом плане очень проигрывает).

Качественные фреймворки, огромнейшее количество учебных материалов, дружелюбное комьюнити, простота кодинга: все это делает Python действительно конкурентным языком, который вряд ли сдаст собственные позиции в наступившем 2019-ом.

• Java

Java – является одним из практичных языком программирования для изучения. Его популярность невозможно переоценить, так как основная масса (90%) компаний из списка Fortune используют Java для разработки бэкэнд-систем и десктопных приложений. Кроссплатформенность достигнута с помощью JVM.

В Java, как и во множестве передовых языков, включая C++, Python, применяется принцип объектно-ориентированного программирования (ООП). Java в основном применяется для создания серверных приложений и мобильных ПО. Также это база нативных приложений под Android. Этот язык очень востребован у разработчиков.

• C++

C++ был разработан в 1983, как альтернатива C, и тут же обрел заслуженную популярность. Его ключевая особенность – предопределенные классы. Microsoft Windows и Google Chrome – самые известные примеры проектов, созданных на C++. Данный список могут пополнить проекты Adobe и Amazon’a. Этот язык программирования остаётся востребованным и на сегодняшний день, так как у него очень мощный набор инструментов, который может быть адаптирован в самых различных сферах, например, финансы, банки, игры, связь, электронные платежные системы, розничная торговля и т.д.

Умение использовать C++ позволяет с легкостью писать игры и сложные коммерческие системы наряду с простыми приложениями. Этот

язык – один из самых универсальных и производительных языков программирования, предоставляющий немало необходимых функций.

• R – язык, который используется для статистической обработки данных и работы с графикой, также это свободная программная среда вычислений с открытым исходным кодом в рамках проекта GNU. R – проект схожий с языком «S» (Bell Labs), альтернативная реализация языка S.

R поддерживает широкий спектр статистических и численных методов и обладает неплохой расширяемостью с помощью пакетов. Сами пакеты – это библиотеки для работы специфических функций или специальных областей применения. В базовую поставку R включен основной набор пакетов, а всего по состоянию на 2006 год доступно более 800 пакетов.

Еще одной особенностью R являются графические возможности – создание качественной графики, которая имеет возможность включать математические символы.

• C#

Созданный компанией Microsoft, мультипарадигмальный язык программирования общего назначения, применялся для разработки приложений на платформе Microsoft. C# – это объектно-ориентированный язык, который применяется в разработке приложений, основанных на .NET frameworks. C# – лучший язык для создания нативных приложений для платформы Microsoft. Кроме того, он является рекомендуемым языком для разработки игр с использованием движка Unity Game.

Приоритетом разработчиков данного языка была его простота, и так как это язык высокого уровня, он больше похож на английский, чем другие. C# позволяет разработчику сосредоточиться на алгоритме, а не на деталях реализации – сложные конструкции в нём заключены в абстракции.

На C# возможно написать что угодно: веб-сервисы, мобильные ПО, серверные приложения и т.д. Также считается что эта платформа упрощает разработку приложений под Android и iOS.

Сравнение языков программирования представлено в таблице 1.

Таблица 1

Сравнение языков программирования

Название	Разработчик	Система программирования	Образовательные возможности
Java	Разработан компанией Sun Microsystems (в последующем приобретённой компанией Oracle) Инженеры компании Sun Microsystems: Патрик Ноутон (Patrick Naughton) – руководитель группы инженеров Джеймс Гослинг (James Gosling) – член Совета директоров. Разработка Java началась в 1990 году, первая официальная версия – Java 1.0, – была выпущена только 21 января 1996 года	Idea, Eclipce, NetBeans, JDeveloper, Android Studio,	Создание приложений, игр и веб-сайтов, системные приложения, приложения для мобильных телефонов
Python	Гвидо ван Россум и Python Software Foundation . Первая стабильная версия появилась в январе 1994 года	Python ide, PyCharm, Visual Studio, Spyder, Thonny	Создание приложений, игр и веб-сайтов, математические расчеты, нейросети, системные приложения,

			операционные системы
C++	Появился в 1983 г. Автор Страуструп, Бьёрн	Visualstudio, Eclipse CDT, NetBeans, CodeLite, Code::Blocks	Создание приложений, игр, математические расчеты, нейросети
C#	Microsoft, Хейлсберг, Андерс, Появился в 2000	Visualstudio, Project Rider, Eclipse, MonoDevelop, Code::Blocks,	Создание приложений, игр и веб-сайтов
R	Росс Айхэка Роберт Джентлмен Появился в 1993	RStudio, Revolution Open, CRAN Repository, Visual Studio 2015	Математические расчёты, создание графики, документирование статистических обсчетов, создание интерактивных веб-приложений

В ходе сравнения языков программирования, выявили самый подходящий для глубокого изучения в школе – Python. На нём можно разработать что угодно, огромное количество библиотек и возможность писать простой и понятный код.

1.3. Языки программирования, изучаемые в школе

К рассмотрению возьмем старшие классы. Из рекомендованных учебников 10-11 классов, какие языки программирования изучаются, а также используемые в ЕГЭ и ОГЭ [23].

- Босова Л. Л., Босова А. Ю. ООО «Бином. Лаборатория знаний» – Информатика. Базовый уровень. 11 класс.

Глава 2 посвящена алгоритмам и элементам программирования. Основные сведения об алгоритмах. Алгоритмические структуры. Запись

алгоритмов на языках программирования, рассматривается программирование на Pascal. Структурированные типы данных. Структурное программирование.

- Гейн А. Г., Гейн А. А. АО Издательство «Просвещение» – Информатика. Базовый уровень. 10 класс. Рассматриваются основы языка программирования, алгоритмов, основные понятия.
- Гейн А. Г., Сенокосов А. И. АО Издательство «Просвещение» – Информатика. Базовый уровень. 11 класс. Задания для подготовки к ЕГЭ, языки программирования Basic, Pascal.
- Поляков К. Ю., Еремин Е. А. ООО «Бином. Лаборатория знаний» – Информатика. Базовый и углубленный уровни. (в двух частях), 10класс. Рассматриваются понятия языка программирования. В первой части есть примеры кода на языке Pascal, во второй обучение программированию на Pascal.
- Поляков К. Ю., Еремин Е. А. ООО «Бином. Лаборатория знаний» – Информатика. Базовый и углубленный уровни. (в двух частях), 11 класс. Во второй части программирование и примеры на языке Pascal.
- Угринович Н. Д. ООО «Бином. Лаборатория знаний» – Информатика. Базовый уровень. 10 класс. Основы Visual Basic, Visual C#. Краткая история развития языков программирования, Assembler, Fortran, Cobol, Basic, Pascal, C, C++, Object Pascal, QBasic, Java, JavaScript.
- Угринович Н. Д. ООО «Бином. Лаборатория знаний» – Информатика. Базовый уровень. 10 класс. В главе 5, подготовка к ЕГЭ – языки программирования VisualBasic, Visual C#, Visual J#, Pascal
- Калинин И. А., Самылкина Н. Н. ООО «Бином. Лаборатория знаний» – Информатика. Углубленный уровень. 10 класс. Рассматриваются Pascal, C.
- Семакин И. Г., Хеннер Е.К., Шестакова Л. В. ООО «Бином. Лаборатория знаний» – Информатика. Углубленный уровень, (в двух частях). 10 класс. В части 1 и 2 программирование на Pascal.

- Семакин И. Г., Хеннер Е.К., Шестакова Л. В. ООО «Бином. Лаборатория знаний» – Информатика. Углубленный уровень, (в двух частях). 11 класс. Программирование на Pascal.

На основе рассмотренных учебников, можно сделать вывод, что в старшей школе рассматриваются языки программирования Pascal, Basic, Python, Visual Basic, Assembler, Fortran, Cobol, C, C++, Object Pascal, QBasic, Java, JavaScript, Visual C#, Visual J#. Более углубленные знания ученики получают об языке программирования Pascal.

- Задания ОГЭ 2018, языки программирования, пример из заданий по языкам программирования (К. Поляков) [22].

B8 – Оператор присваивания (Pascal) (рис. 1).

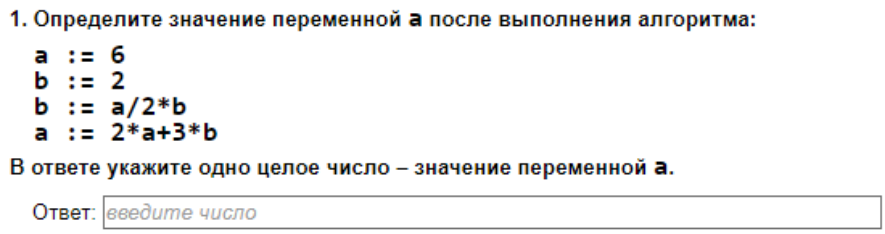


Рис. 1. Задание на Pascal

B8 – Оператор присваивания (Python) (рис. 2).

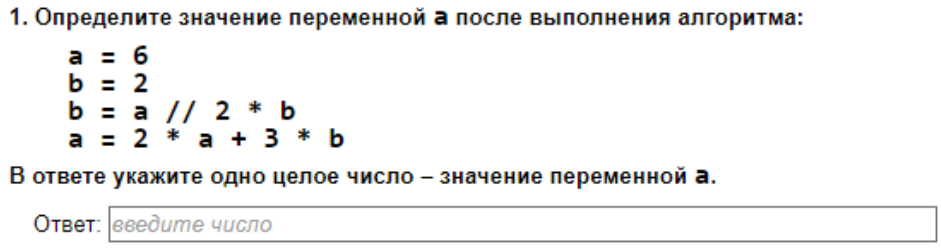


Рис. 2. Задание на Python

B8 – Оператор присваивания (C, C++) (рис. 3).

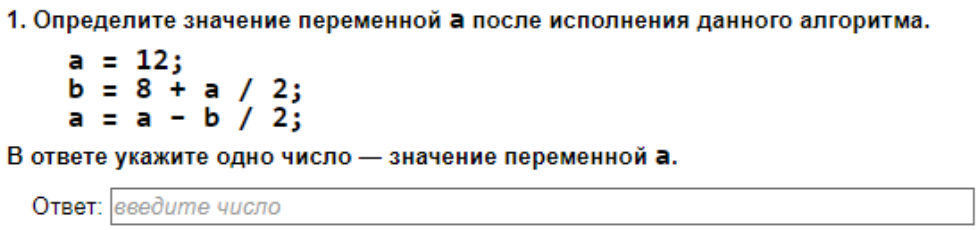


Рис. 3. Задание на C, C++

В ОГЭ встречаются задания с языками программирования Pascal, Python, C, C++.

- Задания ЕГЭ 2018, пример из заданий по языкам программирования. [25]

B8 – Задания включают языки программирования – Basic, Python, Pascal, C++ и алгоритмический язык (рис. 4). Так же эти языки встречаются в других заданиях части А и В.

8. Запишите число, которое будет напечатано в результате выполнения следующей программы. Для Вашего удобства программа представлена на пяти языках программирования.

Бейсик

```
DIM S, N AS INTEGER
S = 0
N = 0
WHILE S < 125
  S = S + 8
  N = N + 2
WEND
PRINT N
```

Python

```
s = 0
n = 0
while s < 125:
    s = s + 8
    n = n + 2
print(n)
```

Алгоритмический язык

```
алг
нач
  цел n, s
  n := 0
  s := 0
  нц пока s < 125
    s := s + 8
    n := n + 2
  кц
  вывод n
кон
```

Паскаль

```
var s, n: integer;
begin
  s := 0;
  n := 0;
  while s < 125 do
  begin
    s := s + 8;
    n := n + 2
  end;
  writeln(n)
end.
```

Си++

```
#include <iostream>
using namespace std;
int main()
{  int s = 0, n = 0;
   while (s < 125) { s = s + 8; n = n + 2; }
   cout << n << endl;
   return 0;
}
```

Рис. 4. Basic, Python, Pascal, C++ и алгоритмический язык

Выводы по Главе 1

В теоретической части рассмотрено понятие языка программирования, несколько классификаций языков программирования, сравнили наиболее популярные из них. Определили основные критерии, по которым выбирается язык программирования для изучения в школе. Также были рассмотрены учебные материалы, существующие для школ и особенности включения курса «программирование» в школьную программу предмета информатика.

ГЛАВА 2. МЕТОДИЧЕСКИЕ ОСОБЕННОСТИ ИЗУЧЕНИЯ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ В ШКОЛЕ

2.1. Анализ нормативных документов

Проанализировав ФГОС среднего общего образования (10 - 11 кл.) «Приказ Минобрнауки России от 17.05.2012 N 413» по теме моей ВКР можно выделить требования к изучению предметной области информатика:

- сформированность представлений о социальных, культурных и исторических факторах становления математики и информатики;
- сформированность основ логического, алгоритмического и математического мышления;
- сформированность умений применять полученные знания при решении различных задач;
- сформированность представлений о математике как части общечеловеческой культуры, универсальном языке науки, позволяющем описывать и изучать реальные процессы и явления;
- сформированность представлений о роли информатики и ИКТ в современном обществе, понимание основ правовых аспектов использования компьютерных программ и работы в Интернете;
- сформированность представлений о влиянии информационных технологий на 16 жизнь человека в обществе;
- понимание социального, экономического, политического, культурного, юридического, природного, эргономического, медицинского и физиологического контекстов информационных технологий;
- принятие этических аспектов информационных технологий;
- осознание ответственности людей, вовлеченных в создание и использование информационных систем, распространение информации.

В своей работе мы будем разрабатывать курс по выбору «Основы программирования на Python», методическая разработка должна содержать:

- Краткие сведения об авторской методической разработке и авторском компьютерном приложении.
- Сведения о проведенных занятиях и использовании авторского компьютерного приложения.
- Самооценка профессиональных достижений на практике, рекомендации по организации и проведению практики.

Требования к предметным результатам освоения базового курса информатики должны отражать:

- сформированность представлений о роли информации и связанных с ней процессов в окружающем мире;
- владение навыками алгоритмического мышления и понимание необходимости формального описания алгоритмов;
- владение умением понимать программы, написанные на выбранном для изучения универсальном алгоритмическом языке высокого уровня; знанием основных конструкций программирования;
- умением анализировать алгоритмы с использованием таблиц;
- владение стандартными приемами написания на алгоритмическом языке программы для решения стандартной задачи с использованием основных конструкций программирования и отладки таких программ;
- использование готовых прикладных компьютерных программ по выбранной специализации;
- сформированность представлений о компьютерно-математических моделях и необходимости анализа соответствия модели и моделируемого объекта (процесса); способах хранения и простейшей

обработке данных; понятия о базах данных и средствах доступа к ним, умений работать с ними;

- владение компьютерными средствами представления и анализа данных;
- сформированность базовых навыков и умений по соблюдению требований техники безопасности, гигиены и ресурсосбережения при работе со средствами информатизации; понимания основ правовых аспектов использования компьютерных программ и работы в Интернете.

«Информатика» (углубленный уровень) – требования к предметным результатам освоения углубленного курса информатики должны включать требования к результатам освоения базового курса и дополнительно отражать:

- владение системой базовых знаний, отражающих вклад информатики в формирование современной научной картины мира;
- овладение понятием сложности алгоритма, знание основных алгоритмов обработки числовой и текстовой информации, алгоритмов поиска и сортировки;
- владение универсальным языком программирования высокого уровня (по выбору), представлениями о базовых типах данных и структурах данных; умением использовать основные управляющие конструкции;
- владение навыками и опытом разработки программ в выбранной среде программирования, включая тестирование и отладку программ; владение элементарными навыками формализации прикладной задачи и документирования программ;
- сформированность представлений о важнейших видах дискретных объектов и об их простейших свойствах, алгоритмах анализа этих объектов, о кодировании и декодировании данных и причинах искажения данных при передаче; систематизацию знаний, относящихся к

математическим объектам информатики; умение строить математические объекты информатики, в том числе логические формулы;

- владение основными сведениями о базах данных, их структуре, средствах создания и работы с ними;
- владение опытом построения и использования компьютерно-математических моделей, проведения экспериментов и статистической обработки данных с помощью компьютера, интерпретации результатов, получаемых в ходе моделирования реальных процессов; умение оценивать числовые параметры моделируемых объектов и процессов, пользоваться базами данных и справочными системами;
- сформированность умения работать с библиотеками программ; наличие опыта использования компьютерных средств представления и анализа данных [25].

Из приказа № 345 от 28 декабря 2018 г. «О федеральном перечне учебников, рекомендуемых к использованию при реализации имеющих государственную аккредитацию образовательных программ начального общего, основного общего, среднего общего образования» [20].

Учебники, рекомендуемые к использованию при реализации обязательной части основной образовательной программы:

Информатика (базовый уровень):

- Босова Л. Л., Босова А. Ю. ООО «Бином. Лаборатория знаний» – Информатика. Базовый уровень. 11 класс.

Глава 2 посвящена алгоритмам и элементам программирования. Основные сведения об алгоритмах. Алгоритмические структуры. Запись алгоритмов на языках программирования, рассматривается программирование на Pascal. Структурированные типы данных. Структурное программирование [2].

- Гейн А. Г., Гейн А. А. АО Издательство «Просвещение» – Информатика. Базовый уровень. 10 класс. Рассматриваются основы языка программирования, алгоритмов, основные понятия [3].

- Гейн А. Г., Сенокосов А. И. АО Издательство «Просвещение» – Информатика. Базовый уровень. 11 класс. Задания для подготовки к ЕГЭ, языки программирования Basic, Pascal [4].
- Поляков К. Ю., Еремин Е. А. ООО «Бином. Лаборатория знаний» – Информатика. Базовый и углубленный уровни. (в двух частях), 10класс. Рассматриваются понятия языка программирования. В первой части есть примеры кода на языке Pascal, во второй обучение программированию на Pascal. Введение в язык Python [7].
- Поляков К. Ю., Еремин Е. А. ООО «Бином. Лаборатория знаний» – Информатика. Базовый и углубленный уровни. (в двух частях), 11 класс. Во второй части программирование и примеры на языке Pascal[8].
- Угринович Н. Д. ООО «Бином. Лаборатория знаний» – Информатика. Базовый уровень. 10 класс. Основы Visual Basic, Visual C#. Краткая история развития языков программирования, Assembler, Fortran, Cobol, Basic, Pascal, C, C++, Object Pascal, QBasic, Java, JavaScript [15].
- Угринович Н. Д. ООО «Бином. Лаборатория знаний» – Информатика. Базовый уровень. 11 класс. В главе 5, подготовка к ЕГЭ – языки программирования VisualBasic, Visual C#, Visual J#, Pascal [16].

Информатика (углубленный уровень)

- Калинин И. А., Самылкина Н. Н. ООО «Бином. Лаборатория знаний» – Информатика. Углубленный уровень. 10 класс. Рассматриваются Pascal, C [5].
- Семакин И. Г., Хеннер Е.К., Шестакова Л. В. ООО «Бином. Лаборатория знаний» – Информатика. Углубленный уровень, (в двух частях). 10 класс. В части 1 и 2 программирование на Pascal [12].
- Семакин И. Г., Хеннер Е.К., Шестакова Л. В. ООО «Бином. Лаборатория знаний» – Информатика. Углубленный уровень, (в двух частях). 11 класс. Программирование на Pascal [13].

Рассмотрев список рекомендуемых учебников, можно сделать вывод: в 10-11 классах рассматривается в большей степени язык программирования Pascal, современные и востребованные языки рассматриваются в малом объеме, чего недостаточно для подготовки учащихся к программированию на востребованных языках.

Из письма Минобрнауки РФ от 28.10.2015 № 08-1786 «О рабочих программах учебных предметов» выделим требования к разработке курса. Курс должен содержать:

1) пояснительную записку, в которой конкретизируются цели общего образования с учетом специфики учебного предмета;

2) общую характеристику учебного предмета, курса;

3) описание места учебного предмета, курса в учебном плане;

4) личностные, метапредметные и предметные результаты освоения конкретного учебного предмета, курса;

5) содержание учебного предмета, курса.

6) тематическое планирование с определением основных видов учебной деятельности;

7) описание учебно-методического и материально-технического обеспечения образовательной деятельности;

8) планируемые результаты изучения учебного предмета, курса [24].

2.2. Курс по выбору «Основы программирования на Python»

Пояснительная записка

Базового курса информатики недостаточно для полного овладения современным алгоритмическим языком программирования.

У некоторых школьников возникает интерес не только к знакомству с программированием, но и к его углубленному изучению. Как правило, у них есть способности и желание. Кроме того, любовь к программированию многие учителя информатики принесли из своей профессиональной

деятельности, и, конечно же, им хочется передать эту любовь и своим ученикам.

Решаемые задачи адаптируются под уровень математической подготовки учеников. Полноценные занятия можно проводить лишь тогда, когда на уроки информатики отводится не менее двух (спаренных) учебных часов в неделю. В другом случае изучение программирования лучше проводить в рамках факультатива.

Курс по выбору направлен на совершенствование практических навыков работы за компьютером и информационной компетенции учащихся с опорой на знания, полученные на уроках информатики. Уделяется особое внимание алгоритмизации и практическим навыкам программирования.

Курс «Основы программирования на Python» представляет собой вводный курс по программированию, дающий представление о базовых понятиях структурного программирования (данных, операциях, переменных, ветвлениях в программе, циклах и функциях) на языке Python.

Выбор Python обусловлен рядом преимуществ перед другими языками: простота кода, быстрая реализация, пошаговое интерактивное исполнение во время отладки программы.

Цели курса:

Основная цель курса «Основы программирования на языке Python» – формирование базовых понятий структурного программирования, развитие логики обучающихся. Данный курс призван развивать логическое мышление учащихся и аналитический стиль мышления начинающих программистов. Он предназначен для учащихся 10-11 классов. Основой курса должны быть умение самостоятельно составлять алгоритмы решения задач, реализация этих алгоритмов непосредственно в среде разработки на компьютере, а также формирование правильного стиля мышления.

Задачи курса:

– развивать аналитическое и логическое мышление школьников;

– показать практическую значимость программирования для решения задач в различных областях жизнедеятельности человека;

– научить учащихся основам программирования с использованием системы программирования Python;

– научить составлению и оформлению программ в соответствии с требованиями языка программирования Python;

– развить общую информационную культуру, и подготовить к профессиональной деятельности;

Объем курса составляет 17 часов: 1 теоретический, 8 комбинированных и 8 практических уроков.

До изучения:

Учащиеся должны знать из базового курса информатики:

- понятие алгоритма;
- свойства алгоритма;
- формы записи алгоритма;
- язык псевдокода;
- основные алгоритмические структуры;
- правила записи арифметических выражений;

Учащиеся должны уметь:

- составлять алгоритмические структуры при решении задач;
- записывать алгоритмы, не допуская двусмысленной записи;
- составлять алгоритм решения задач и переводить его на язык псевдокода;
- конструировать решение задачи из минимального числа инструкций;
- записывать вспомогательные алгоритмы в виде подпрограмм.

По завершению:

Учащиеся должны знать:

- среду программы Geany;
- типы данных языка Python;
- встроенные стандартные функции;

Учащиеся должны уметь:

– переводить арифметическую запись выражений на язык программирования;

– использовать функции модулей math и cmath для решения задач;

– определять типы данных;

– записывать логические выражения;

– использовать в задачах такие конструкции как: цикл и ветвление.

– выполнять запись программы на языке Python, тестирование и отладку программы;

– решать задачи по теме: строки, списки и словари.

Рекомендуемые требования к аппаратному и программному обеспечению:

Для успешного проведения практикума по программированию на Python на рабочих местах должны быть установлены:

– Python (версия не ниже 3.5)

– модули Tkinter и NumPy

– среды разработки на Python: IDLE, Eric или Geany, а также какие-либо эмуляторы терминалов _ xterm, rxvt и т.п.

В сборке от ALT Linux следует проверить наличие в системе следующих пакетов:

– geany;

– eric;

– xterm;

– python;

– python-base;

– python-doc;

– python-module-numpy;

– python-modules;

– python-modules-encodings;

– python-modules-tkinter;

– python-tools-idle.

Некоторые из перечисленных пакетов будут установлены по зависимостям при установке Python, Eric и Geany с помощью менеджера пакетов, остальные нужно установить вручную.

При создании программ удобно одновременно видеть текст программы и результаты её выполнения. Кроме того, бывает полезно выполнять программу по шагам и при этом следить за значениями каких-то переменных. Все эти возможности реализуются в так называемых интегрированных средах разработки (Integrated Development Environment, IDE).

Современные IDE, входящие в дистрибутивы Linux, могут работать с разными языками программирования. Существует IDE, лучше всего приспособленные для работы с одним конкретным языком, которые с другими языками работают, так сказать, факультативно. Кроме того, существуют IDE, которые одинаково успешно обеспечивают работу с самыми разными языками, как в режиме интерпретатора, так и в режиме компилятора.

В зависимости от версии ALT Linux удобно пользоваться либо Geany, либо Eric. На практических занятиях рекомендуется рассмотреть особенности работы в обоих IDE.

Таблица 2

Тематическое планирование курса

№	Тема урока	Количество часов		
		Всего	**Теория**	**Практика**
1	Введение в язык программирования Python.	1	1	-
2	Типы данных. Определение переменной. Ввод и вывод данных. Среда Geany и PyScripter для написания программ на языке Python.	2	1	1
3	Решение вычислительных задач.	1	–	1
4	Решение вычислительных задач на основе встроенных функций.	1	–	1
5	Решение задач на основе функций и констант, определенных в модулях math и cmath.	1	–	1
6	Условный оператор полной и неполной формы. Логические выражения.	2	1	1
7	Множественное ветвление.	1	1	–
8	Решение задач на применение условного оператора.	1	–	1
9	Цикл While.	1	1	–
10	Цикл For.	1	1	–
11	Циклы.	1	–	1
12	Строки.	1	1	–
13	Списки.	1	1	–
14	Словари.	1	1	–
15	Контрольная работа.	1	–	1
Итого:	17			

Три полных конспекта уроков предоставлены в текстовом варианте в Приложении.

Поурочное планирование курса

Урок 1. Введение в язык программирования Python

Тип урока: урок усвоения новых знаний.

Цель урока: познакомить учащихся с особенностями языка программирования Python, изучить основы программирования на данном языке.

Задачи урока:

Образовательная: сформировать представление о языке Python.

Развивающая: развивать у учащихся логическое и алгоритмическое мышление, навыки мыслительной деятельности, включая каждого учащегося в учебно-познавательный процесс и создавая условия для работы каждого в индивидуальном психологическом ритме.

Воспитательная: способствовать формированию самооценки (саморефлексии).

Основные понятия: интерпретатор, интерактивный режим, среда разработки IDLE.

Методическая рекомендация: в начале урока учитель предлагает ученикам открыть на компьютере папку с подготовленной презентацией, теоретическую часть. и открывает у себя на компьютере, демонстрирует с помощью проектора. В ходе урока ученики повторяют действия, которые учитель демонстрирует для закрепления знаний. В конце урока учитель задает вопросы по пройденному материалу.

Вопросы для контроля:

1. Два варианта как начать писать и как запускать код Python.
2. Как сохранить код в файл?

Урок 2. Типы данных. Определение переменной. Ввод и вывод данных. Среда Geany Geany и PyScripter для написания программ на языке Python.

Тип урока: урок усвоения новых знаний.

Цель урока: познакомить учащихся с типами данных, вводом и выводом данных, с переменными, со средой Geany Geany и PyScripter для написания программ на языке Python.

Задачи урока:

Образовательная: сформировать представление о языке Python, научиться писать простой код на языке программирования Python в среде разработки.

Развивающая: развивать у учащихся логическое и алгоритмическое мышление, навыки мыслительной деятельности, включая каждого учащегося в учебно-познавательный процесс и создавая условия для работы каждого в индивидуальном психологическом ритме.

Воспитательная: воспитывать у учащихся самостоятельность, активность, интерес к предмету.

Основные понятия: тип данных, ввод, вывод, переменная, операнд, оператор.

Методические рекомендации: в начале урока учитель предлагает ученикам открыть на компьютере папку с подготовленной презентацией, теоретическую часть, и открывает у себя на компьютере, демонстрирует с помощью проектора. В ходе урока ученики повторяют действия, которые учитель демонстрирует для закрепления знаний. В конце урока учитель дает задание для самостоятельной работы и задает вопросы по пройденному материалу.

Вопросы для контроля:

1. Какие типы данных сегодня изучили?
2. Как осуществлять ввод и вывод данных?
3. Как задать переменную?

Урок 3. Решение вычислительных задач.

Тип урока: урок закрепления новых знаний.

Цель урока: научить учащихся самостоятельно решать вычислительные задачи.

Задачи урока:

Образовательная: научить решать элементарные задачи в Python.

Развивающая: развивать у учащихся логическое и алгоритмическое мышление, навыки мыслительной деятельности, включая каждого учащегося в учебно-познавательный процесс и создавая условия для работы каждого в индивидуальном психологическом ритме.

Воспитательная: воспитывать у учащихся самостоятельность, активность, интерес к предмету.

Основные понятия: тип данных, ввод, вывод, переменная, операнд, оператор.

Методические рекомендации: в начале урока учитель показывает пример выполнения заданий и выдает задания для самостоятельной работы. В ходе выполнения заданий учитель помогает и отвечает на вопросы учеников.

Вопросы для контроля:

1. Какие типы данных вы знаете?
2. Как осуществлять ввод и вывод данных?
3. Как задать переменную?

Урок 4. Решение вычислительных задач на основе встроенных функций.

Тип урока: комбинированный урок усвоения новых знаний.

Цель урока: научить учащихся решать вычислительные задачи на основе встроенных функций.

Задачи урока:

Образовательная: научить решать задачи в Python.

Развивающая: развивать у учащихся логическое и алгоритмическое мышление, навыки мыслительной деятельности, включая каждого

учащегося в учебно-познавательный процесс и создавая условия для работы каждого в индивидуальном психологическом ритме.

Воспитательная: воспитывать у учащихся самостоятельность, активность, интерес к предмету.

Основные понятия: функция, цикл.

Методическая рекомендация: в начале урока учитель предлагает ученикам открыть на компьютере папку с подготовленной презентацией, теоретическую часть, и открывает у себя на компьютере, демонстрирует с помощью проектора. В ходе урока ученики выполняют задания вместе с учителем, для закрепления знаний.

Вопросы для контроля:

1. Что такое функция?
2. Как записать цикл?
3. Для чего нужна команда def?

Урок 5. Решение задач на основе функций и констант, определенных в модулях math и cmath.

Тип урока: комбинированный урок усвоения новых знаний.

Цель урока: научить учащихся решать вычислительные задачи на основе функций и констант, определенных в модулях math и cmath.

Задачи урока:

Образовательная: научить применять на практике модули math и cmath.

Развивающая: развивать у учащихся логическое и алгоритмическое мышление, навыки мыслительной деятельности, включая каждого учащегося в учебно-познавательный процесс и создавая условия для работы каждого в индивидуальном психологическом ритме.

Воспитательная: воспитывать у учащихся самостоятельность, активность, интерес к предмету.

Основные понятия: модуль Math, модуль Cmath.

Методическая рекомендация: в начале урока учитель предлагает ученикам открыть на компьютере папку с подготовленной презентацией, теоретическую часть, и открывает у себя на компьютере, демонстрирует с помощью проектора. В ходе урока ученики выполняют задания вместе с учителем, для закрепления знаний.

Вопросы для контроля:

1. Что представляет собой модуль math и Cmath?

Урок 6. Условный оператор полной и неполной формы. Логические выражения.

Тип урока: урок усвоения и применения новых знаний.

Цель урока: познакомить учащихся с условным оператором полной и неполной формы. Научить составлять логические выражения.

Задачи урока:

Образовательная: учащиеся должны научиться составлять логические выражения.

Развивающая: развивать у учащихся логическое и алгоритмическое мышление, навыки мыслительной деятельности, включая каждого учащегося в учебно-познавательный процесс и создавая условия для работы каждого в индивидуальном психологическом ритме.

Воспитательная: воспитывать у учащихся самостоятельность, активность, интерес к предмету.

Основные понятия: Логический оператор, блок-схема, if, else.

Методическая рекомендация: в начале урока учитель рассказывает теоретическую часть и выдает задания для самостоятельной работы. В ходе выполнения заданий учитель помогает и отвечает на вопросы учеников.

Вопросы для контроля:

1. Что такое условный оператор?

Урок 7. Множественное ветвление.

Тип урока: урок усвоения новых знаний.

Цель урока: познакомить учащихся с множественным ветвлением.

Задачи урока:

Образовательная: познакомить учащихся с множественным ветвлением, научить применять на практике.

Развивающая: развивать у учащихся логическое и алгоритмическое мышление, навыки мыслительной деятельности, включая каждого учащегося в учебно-познавательный процесс и создавая условия для работы каждого в индивидуальном психологическом ритме.

Воспитательная: воспитывать у учащихся самостоятельность, активность, интерес к предмету.

Основные понятия: ветвление.

Методическая рекомендация: в начале урока учитель рассказывает теоретическую часть и выдает задания для практической части. В ходе выполнения заданий учитель помогает и отвечает на вопросы учеников.

Вопросы для контроля:

1. Как организовать множественное ветвление?

Урок 8. Решение задач на применение условного оператора.

Тип урока: урок применения полученных знаний.

Цель урока: научить решать задачи на применение условного оператора.

Задачи урока:

Образовательная: научить решать задачи на применение условного оператора.

Развивающая: развивать у учащихся логическое и алгоритмическое мышление, навыки мыслительной деятельности, включая каждого учащегося в учебно-познавательный процесс и создавая условия для работы каждого в индивидуальном психологическом ритме.

Воспитательная: воспитывать у учащихся самостоятельность, активность, интерес к предмету.

Основные понятия: условный оператор.

Методическая рекомендация: в начале урока задает вопросы по материалу из прошлого урока и выдает задания для практической части. В ходе выполнения заданий учитель помогает и отвечает на вопросы учеников.

Урок 9. Цикл While.

Тип урока: урок усвоения новых знаний.

Цель урока: познакомить учащихся с циклом While.

Задачи урока:

Образовательная: научить составлять циклы While на языке программирования Python.

Развивающая: развивать у учащихся логическое и алгоритмическое мышление, навыки мыслительной деятельности, включая каждого учащегося в учебно-познавательный процесс и создавая условия для работы каждого в индивидуальном психологическом ритме.

Воспитательная: воспитывать у учащихся самостоятельность, активность, интерес к предмету.

Основные понятия: цикл.

Методическая рекомендация: в начале урока учитель рассказывает теоретическую часть и выдает задания для практической части. В ходе выполнения заданий учитель помогает и отвечает на вопросы учеников.

Вопросы для контроля:

1. Что такое цикл?
2. Приведите пример цикла в реальной жизни.

Урок 10. Цикл For.

Тип урока: урок усвоения новых знаний.

Цель урока: познакомить учащихся с циклом For.

Задачи урока:

Образовательная: научить составлять циклы For на языке программирования Python.

Развивающая: развивать у учащихся логическое и алгоритмическое мышление, навыки мыслительной деятельности, включая каждого учащегося в учебно-познавательный процесс и создавая условия для работы каждого в индивидуальном психологическом ритме.

Воспитательная: воспитывать у учащихся самостоятельность, активность, интерес к предмету.

Основные понятия: цикл.

Методическая рекомендация: в начале урока учитель рассказывает теоретическую часть и выдает задания для практической части. В ходе выполнения заданий учитель помогает и отвечает на вопросы учеников.

Вопросы для контроля:

1. Приведите пример использования цикла For.

Урок 11. Лабораторная работа по теме: Циклы.

Тип урока: урок применения знаний, полученных на предыдущих занятиях.

Цель урока: научиться самостоятельно составлять циклы.

Задачи урока:

Образовательная: научить составлять циклы на языке программирования Python.

Развивающая: развивать у учащихся логическое и алгоритмическое мышление, навыки мыслительной деятельности, включая каждого учащегося в учебно-познавательный процесс и создавая условия для работы каждого в индивидуальном психологическом ритме.

Воспитательная: воспитывать у учащихся самостоятельность, активность, интерес к предмету.

Основные понятия: цикл.

Методическая рекомендация: в начале урока задает вопросы по материалу из прошлого урока и выдает задания для практической части. В ходе выполнения заданий учитель помогает и отвечает на вопросы учеников.

Вопросы для контроля:

1. Что такое цикл for?
2. Что такое цикл while?

Урок 12. Строки.

Тип урока: урок усвоения новых знаний.

Цель урока: познакомить учащихся с понятием строки.

Задачи урока:

Образовательная: познакомить учащихся с основными понятиями, научить применять на практике.

Развивающая: развивать у учащихся логическое и алгоритмическое мышление, навыки мыслительной деятельности, включая каждого учащегося в учебно-познавательный процесс и создавая условия для работы каждого в индивидуальном психологическом ритме.

Воспитательная: воспитывать у учащихся самостоятельность, активность, интерес к предмету.

Основные понятия: строка, конкатенация, дублирование, индекс, подстрока.

Методическая рекомендация: в начале урока учитель рассказывает теоретическую часть и выдает задания для практической части. В ходе выполнения заданий учитель помогает и отвечает на вопросы учеников.

Вопросы для контроля:

1. Что такое строки?
2. Что такое конкатенация?
3. Что такое индекс?

Урок 13. Списки.

Тип урока: урок усвоения новых знаний.

Цель урока: познакомить учащихся с основными понятиями.

Задачи урока:

Образовательная: познакомить учащихся с основными понятиями, научить применять на практике.

Развивающая: развивать у учащихся логическое и алгоритмическое мышление, навыки мыслительной деятельности, включая каждого учащегося в учебно-познавательный процесс и создавая условия для работы каждого в индивидуальном психологическом ритме.

Воспитательная: воспитывать у учащихся самостоятельность, активность, интерес к предмету.

Основные понятия: списки.

Методическая рекомендация: в начале урока учитель рассказывает теоретическую часть и выдает задания для практической части. В ходе выполнения заданий учитель помогает и отвечает на вопросы учеников.

Вопросы для контроля:

1. Из чего могут состоять списки?
2. Что такое списки?
3. Как связать списки с переменными?

Урок 14. Словари.

Тип урока: урок усвоения новых знаний.

Цель урока: познакомить учащихся с основными понятиями.

Задачи урока:

Образовательная: познакомить учащихся с основными понятиями, научить применять на практике.

Развивающая: развивать у учащихся логическое и алгоритмическое мышление, навыки мыслительной деятельности, включая каждого

учащегося в учебно-познавательный процесс и создавая условия для работы каждого в индивидуальном психологическом ритме.

Воспитательная: воспитывать у учащихся самостоятельность, активность, интерес к предмету.

Основные понятия: словарь.

Методическая рекомендация: в начале урока учитель рассказывает теоретическую часть и выдает задания для практической части. В ходе выполнения заданий учитель помогает и отвечает на вопросы учеников. В конце урока выдает вопросы для подготовки к контрольной работе.

Вопросы для контроля:

1. Что такое словарь?
2. С помощью чего определяется словарь?
3. Какой тип данных у словарей?

Урок 15. Контрольная работа.

Тип урока: урок самостоятельно применять знания, полученные при изучении курса.

Цель урока: закрепить изученный материал и подвести итоги.

Задачи урока:

Образовательная: проверить умения и знания, полученные на изучении данного курса.

Развивающая: развивать у учащихся логическое и алгоритмическое мышление, навыки мыслительной деятельности, включая каждого учащегося в учебно-познавательный процесс и создавая условия для работы каждого в индивидуальном психологическом ритме.

Воспитательная: воспитывать у учащихся самостоятельность, активность, интерес к предмету.

Методическая рекомендация: ученики выполняют самостоятельно задания при помощи компьютера и тестирования.

2.3. Программно-методическая поддержка курса

В качестве программно-методической поддержки курса по выбору «Основы программирования на Python» был разработан сайт с помощью wix.com. Учебное пособие располагается по адресу https://alenadobryak74.wixsite.com/python.

На рисунке 5 представлена главная страница программно-методической поддержки курса.

Рис. 5. Главная страница

Главная страница содержит основные сведения о том, для чего нужно изучать Python в старшей школе.

В разделе «О проекте» (рис. 6) находится информация о содержании данного сайта.

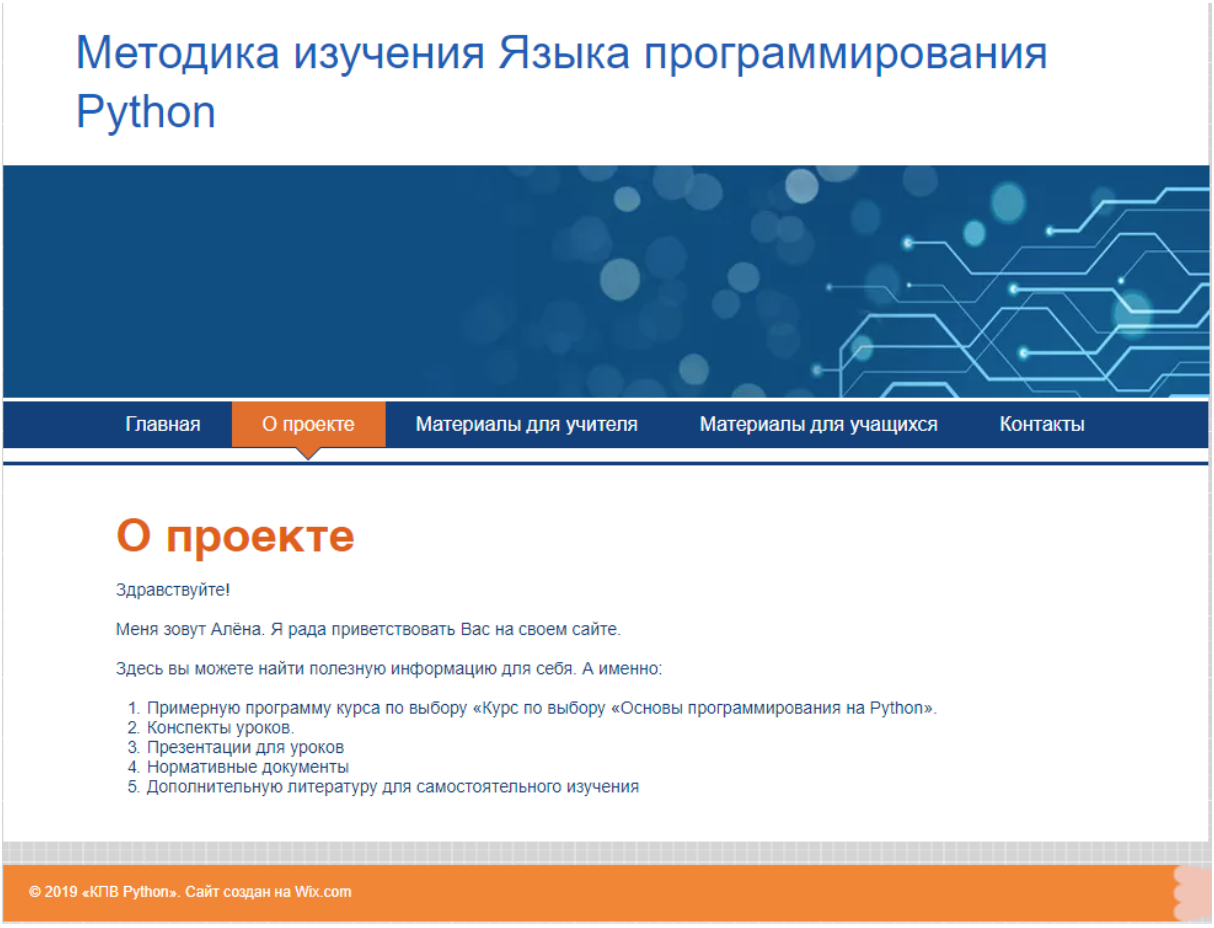


Рис. 6. раздел «О проекте»

В следующем разделе «Материалы для учителя» представлены примерная программа уроков, конспекты уроков в хронологическом порядке и нормативные документы (рис.7).

Рис. 7. Раздел «Материалы для учителя»

В разделе «Материалы для учащихся» расположены презентации к урокам курса (рис. 8) и дополнительная литература для самостоятельного дальнейшего изучения языка программирования Python (рис. 9).

Рис. 8. Презентации к урокам

Рис. 9. Литература для дальнейшего самостоятельного изучения

В разделе «Контакты» представлена информация для организации связи с разработчиком сайта (рис. 10).

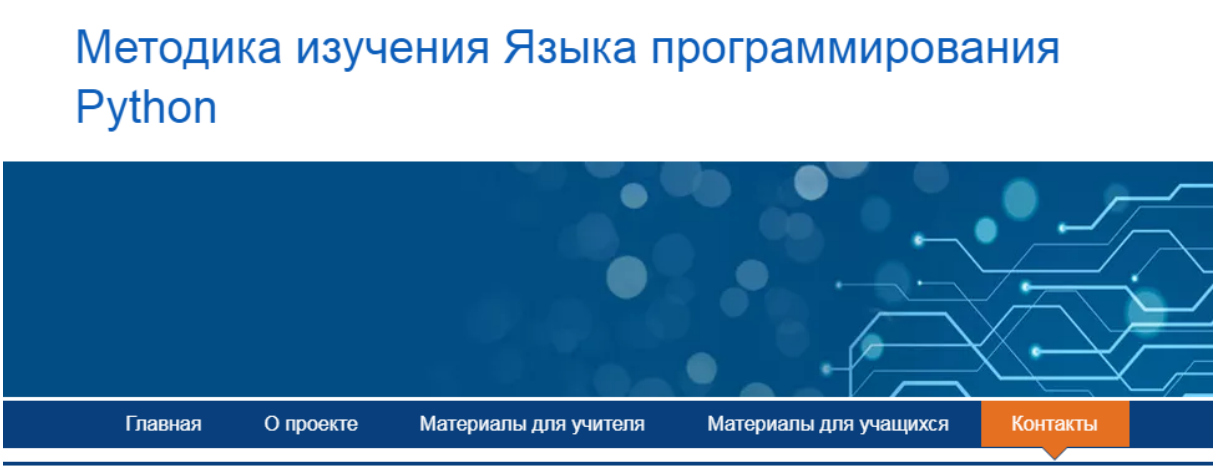


Рис. 10. Контакты

2.4. Апробация результатов исследования в школе

Педагогическая апробация проводилась в рамках научно-исследовательской практики в ГБПОУ «Южно-Уральский государственный колледж», г. Челябинска. Курс изучался у учеников 10-11 класса. В течение 15 занятий была рассмотрена тема программирования на языке Python.

Апробация прошла успешно. Способствовала этому правильная мотивация, цели и задачи изучения темы.

Тема курса оказалась знакома для учащихся, и они быстро включились в работу. В таком возрасте дети особенно заинтересованы в получении знаний, особенно если информация наглядная и простая.

Выводы по Главе 2

В ходе практической части, был проведен анализ нормативных документов. Рассмотрен современный язык программирования Python как наиболее удобный и подходящий в образовательных целях. Составлен курс по выбору Python для учеников заинтересованных в углубленном изучении программирования. Составлены требования для учащихся, допущенных к этому курсу. Обозначены цели и задачи, а так же описаны знания и умения, полученные после прохождения этого курса.

ЗАКЛЮЧЕНИЕ

В ходе выполнения дипломной работы была изучена литература по языкам программирования, проведен анализ современных языков программирования. Было выбрано для анализа пять популярных языков – программирования R, Python, C#, C++, Pascal и Java, проведено сравнение этих языков, выявлены образовательные возможности. Проанализированы языки программирования, изучаемые в школе, на основе стандартов, учебников, программ, ЕГЭ. Предложена методика изучения одного из современных языков программирования в школе Python, план изучения и 15 уроков.

Цель достигнута: рассмотрены методические особенности изучения современных языков программирования в школе и разработан курс «Основы программирования на Python» и программно-методическая поддержка к нему.

Задачи решены: d теоретической части рассмотрено понятие языка программирования Python; рассмотрены современные языки программирования; рассмотрены языки программирования изучаемые в школе; сделаны выводы по изученной информации. В практической части проведен анализ нормативных документов; разработан курс по выбору «Основы программирования на Python»; Разработана программно-методическая поддержка курса в виде электронного пособия.

Гипотеза подтверждена.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Бизли Д. М. Язык программирования Python. Справочник / Бизли Д. М, Г. Ван Россум Босова Л. Л. Босова А. Ю. – М.: ДиаСофт, 2000. – 858 с.

2. Босова Л. Л. Информатика. Базовый уровень. 11 класс / Босова Л. Л. Босова А. Ю. – М.:«Бином. Лаборатория знаний», 2016. – 256 с.

3. Гейн А. Г. Информатика. Базовый уровень. 10 класс / Гейн А. Г. Гейн А. А. – М.: «Просвещение», 2012. – 272 с.

4. Гейн А. Г. Информатика. Базовый уровень. 11 класс. / Гейн А. Г., Сенокосов А. И. – М.:Издательство «Просвещение», 2009. – 336 с.

5. Калинин И. А. Информатика. Углубленный уровень. 10 класс. / Калинин И. А. Самылкина Н. Н. – М.:«Бином. Лаборатория знаний», 2013. – 256 с.

6. Лутц М. Программирование на Python. (пер. с англ.) / Лутц М. – СПб.: Символ-Плюс, 2002. – 1136 с.

7. Поляков К. Ю. Информатика. Базовый и углубленный уровни. (в двух частях), 10класс. / Поляков К. Ю. Еремин Е. А. – М.:«Бином. Лаборатория знаний», 2013. – 344 с.

8. Поляков К. Ю. Информатика. Базовый и углубленный уровни. (в двух частях) 11 класс. / Поляков К. Ю. Еремин Е. А. – М.:«Бином. Лаборатория знаний», 2013. – 304 с.

9. Сузи Р.А. Python. Наиболее полное руководство. / Сузи Р.А. – СПб.: БХВ-Петербург, 2002. – 759 с.

10. Сузи Р.А. Язык программирования Python. / Сузи Р.А. – М: Бином. Лаборатория знаний, 2006. – 328 с.

11. Саммерфилд М. Программирование на Python 3. Подробное руководство (пер. с англ.) / Саммерфилд М. – СПб: Символ-Плюс, 2009. – 608 с.

12. Семакин И. Г. Информатика. Углубленный уровень, (в двух частях) 10 класс / Семакин И. Г. Хеннер Е. К. Шестакова Л. В. – М.: «Бином. Лаборатория знаний», 2017. – 440 с.

13. Семакин И. Г. Информатика. Углубленный уровень (в двух частях), 11 класс. / Семакин И. Г. Хеннер Е.К., Шестакова Л. В. – М.:«Бином. Лаборатория знаний», 2017. – 392 с.

14. Угринович Н.Д. Информатика: Учебник для 10-11 класса. / Угринович Н.Д. – М.: «Бином», 2009. – 512 с

15. Угринович Н. Д. Информатика. Базовый уровень. 10 класс. / Угринович Н. Д. – М.: «Бином. Лаборатория знаний», 2016 – 288с.

16. Угринович Н. Д. Информатика. Базовый уровень. 11 класс. / Угринович Н. Д. – М.:«Бином. Лаборатория знаний», 2016. – 272 с.

17. Фридланд А. Я. Информатика и компьютерные технологии. Основные термины. Толковый словарь. / Фридланд А. Я. Ханамирова Л. С. Фридланд И. А. – СПб.: Астрель, 2003. – 272 с.

18. Хахаев И.А. Практикум по алгоритмизации и программированию на Python. / Хахаев И.А. – М.:Альт Линукс, 2010. – 126 с.

19. Единая коллекция цифровых образовательных ресурсов. [Электронный ресурс]. – http://school-collection.edu.ru

20. Банк документов министерства образования Российской Федерации. [Электронный ресурс]. – https://docs.edu.gov.ru

21. Яkласс – цифровой образовательный ресурс для школ. [Электронный ресурс]. – https://www.yaklass.ru/p/informatika

22. Методические материалы и программное обеспечение. [Электронный ресурс]. – http://kpolyakov.spb.ru.

23. Письмо Минобрнауки РФ от 28.10.2015 № 08-1786 "О рабочих программах учебных предметов – http://lbz.ru/metodist/content/files/pismo-08-1786.pdf.pdf

24. Лешицер В.Р. ЕГЭ 2019. Информатика. Типовые тестовые задания. / Лешицер В.Р.– М.:Издательство "Экзамен", 2019. – 311с.

25. Приказ от 6 октября 2009 г. № 413 об утверждении и введении в действие ФГОС Среднего общего образования – https://fgos.ru/LMS/wm/wm_fgos.php?id=sred

ПРИЛОЖЕНИЕ

Конспект урока № 1

Тема урока: Введение в язык программирования Python

Тип урока: урок усвоения новых знаний.

Цель урока: познакомить учащихся с особенностями языка программирования Python, изучить основы программирования на данном языке.

Задачи урока:

Образовательная: сформировать представление о языке Python

Развивающая: развивать у учащихся логическое и алгоритмическое мышление, навыки мыслительной деятельности, включая каждого учащегося в учебно–познавательный процесс и создавая условия для работы каждого в индивидуальном психологическом ритме;

Воспитательная: способствовать формированию самооценки (саморефлексии).

Этапы урока:

1. Организационный момент – 1 мин.

2. Изучение нового материала – 25 мин.

3. Рефлексия – 2 мин.

4. Закрепление изученного материала – 17 мин.

Оборудование:

- Компьютер, Python (версия не ниже 3.5), модули Tkinter и NumPy, среды разработки на Python: IDLE, Eric или Geany, а также какие-либо эмуляторы терминалов _ xterm, rxvt, проектор.

СТРУКТУРА И ХОД УРОКА

1. Организационный момент 1 мин.

Учитель: *Приветствует класс, проверяет присутствующих.*

Здравствуйте, ребята. Сегодня мы с вами познакомимся с языком программирования Python

2. Изучение нового материала – 25 мин.

Учитель: *Демонстрирует презентацию и рассказывает новый материал*

История языка программирования.

Основным разработчиком ЯП Python является Гвидо ван Россум (Guido van Rossum), первая стабильная версия 1.0 появилась в январе 1994 года. После того как Гвидо разработал

Python, он его выложил в интернет, после этого над его улучшением читало работать целое сообщество программистов. Официальный сайт: http://python.org.

Рассмотрим особенности языка.

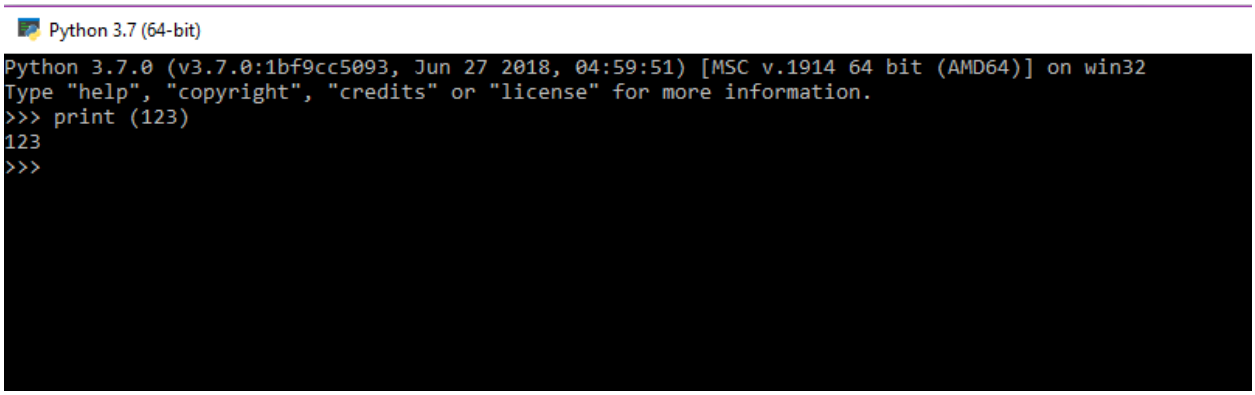
Python – это интерпретируемый ЯП. Python имеет достаточно простой синтаксис. Код на этом языке программирования достаточно легко читается, т.к. используется в нем минимум вспомогательных элементов, а стиль кода жеско продиктован стандартом PEP-8, в котором четко прописано как писать программы.

Python язык высокого уровня: поддерживает объектно-ориентированное программирование. Высокоуровневый – обозначает, что вы будете писать код при помощи самых обычных слов на английском языке, поэтому код будет легко читаться. Также он считается полноценным и универсальным ЯП. На данном ЯП вы сможете разрабатывать что угодно: веб сайты, игры, программы под компьютер, под телефон, различные скрипты, плагины, моды, и. т.д.

Распространяется Python свободно под лицензией подобной GNU General Public License.

Интерактивный режим.

Как уже было сказано интерпретатор выполняет команды построчно, т.е. пишем строку -> интерпретатор выполняет ее -> наблюдаем результат.



Это очень удобно, когда человек только изучает программирование или тестирует какую-нибудь небольшую часть кода. Ведь если работать на компилируемом языке, то пришлось бы сначала написать код на исходном языке программирования, затем скомпилировать и уже потом запустить получившийся файл (с машинным кодом) на исполнение. Если окажется, что где-то в исходном коде была допущена ошибка, то придется перекомпилировать всю программу. Но в интерпретируемых языках нет такой проблемы.

Работать в интерактивном режиме можно в консоли. Для этого следует выполнить команду Python. Запустится интерпретатор, где сначала выведется информация о его версии и иная информация. Далее, приглашение к вводу (>>>).

Задание. Запустите интерпретатор Питона.

Начнем с простого, поскольку никаких команд мы пока не знаем, то будем использовать Питон как калькулятор (возможности языка это позволяют).

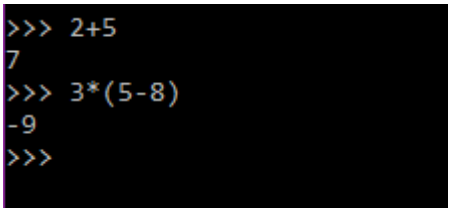


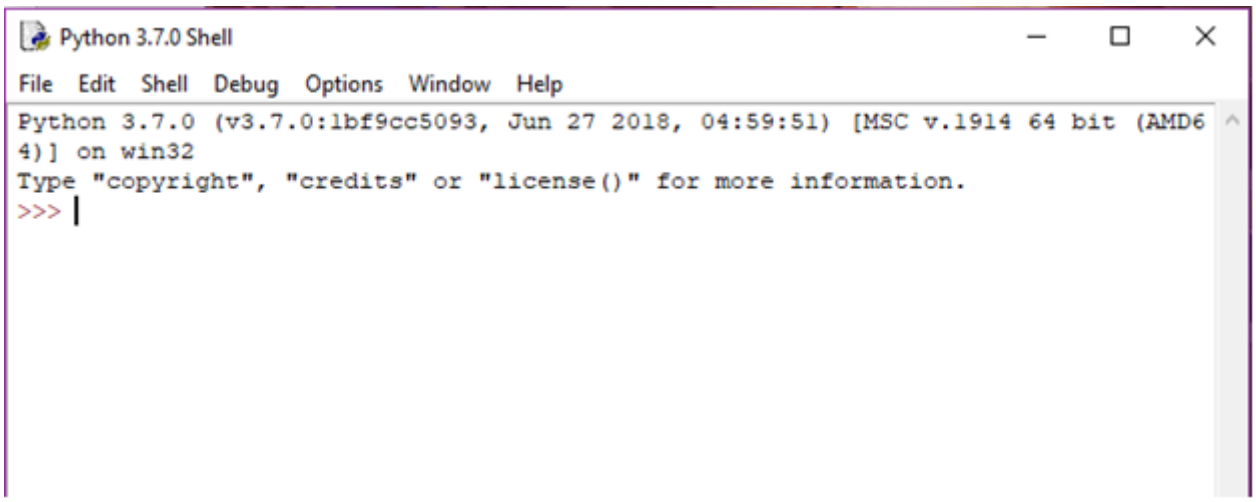
Рис.1. Использование Python в качестве калькулятора

Далее, набирайте подобные примеры в интерактивном режиме (в конце каждого нажимайте Enter).

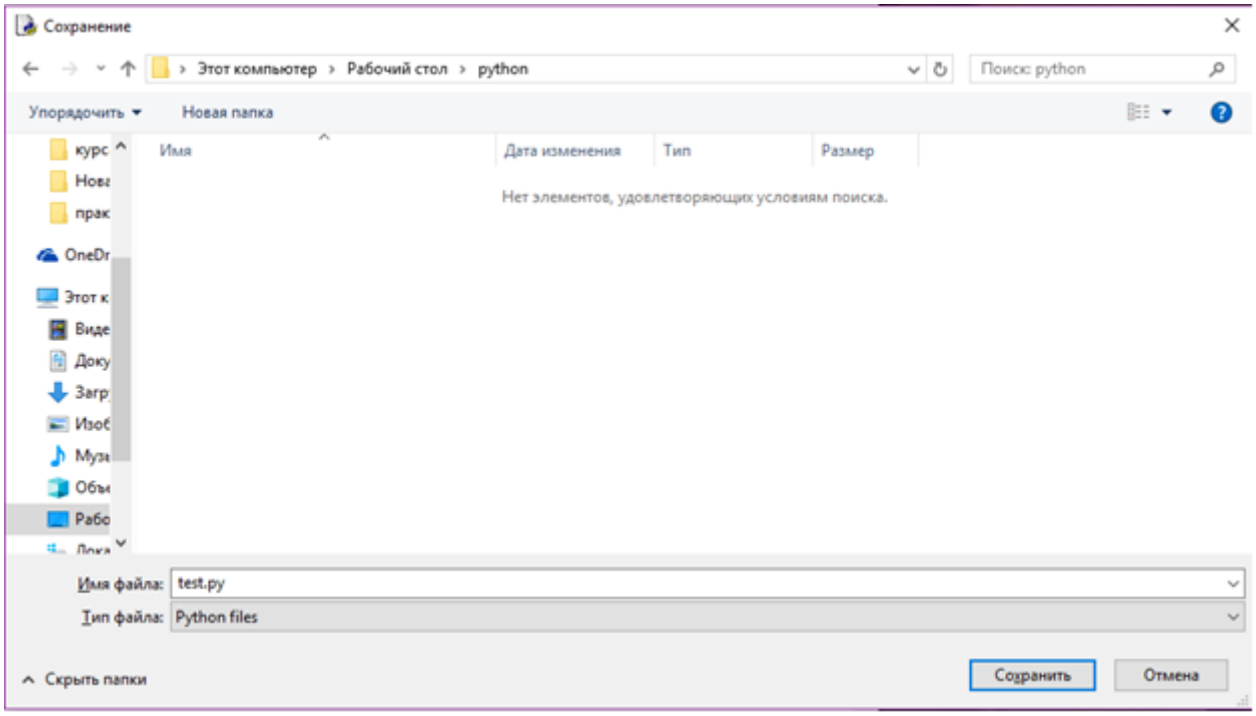
Ответ выдается сразу после нажатия Enter (завершения ввода команды). Бывает, что в процессе ввода допустили ошибку или требуется повторить ранее используемую команду. Чтобы не писать строку сначала, в консоли можно прокручивать список команд, используя для этого стрелки на клавиатуре.

Следующий вариант работы в интерактивном режиме – это работа в среде разработки IDLE, у которой есть интерактивный режим работы. IDLE нужна для того чтобы вбивать прямо в ней какие то команды, на зыке Python. Также IDLE может использоваться и как полноценный редактор кода. В отличие от консольного варианта тут мы можем наблюдать подсветку синтаксиса – все команды выделяются цветом. С помощью комбинаций Alt+N, Alt+P можно прокручивать список команд.

Чтобы запустить IDLE, нажмите на пуск и вбивайте IDLE, запускаем, открывается окошко, это и есть интерактивная оболочка IDLE. Она нужна для того чтобы прямо здесь какие то команды на языке python. Это используют новички для того чтобы тренироваться, смотреть как работают команды, тестировать модули и так далее. IDLE так же может использоваться и как полноценный редактор кода.



Чтобы открыть редактор кода, нужно нажать file – New File. Открывается окошко, в котором вы можете вбивать абсолютно любой код, любой длины в любом количестве. После того как вы вписали весь нужный код, нажмите сохранить, и запустите клавишей F5 или Run – Run Module. Если файл не сохранен, он не сможет запустить и предложит сохранить. Название можно указать любое, обязательно с расширением .py

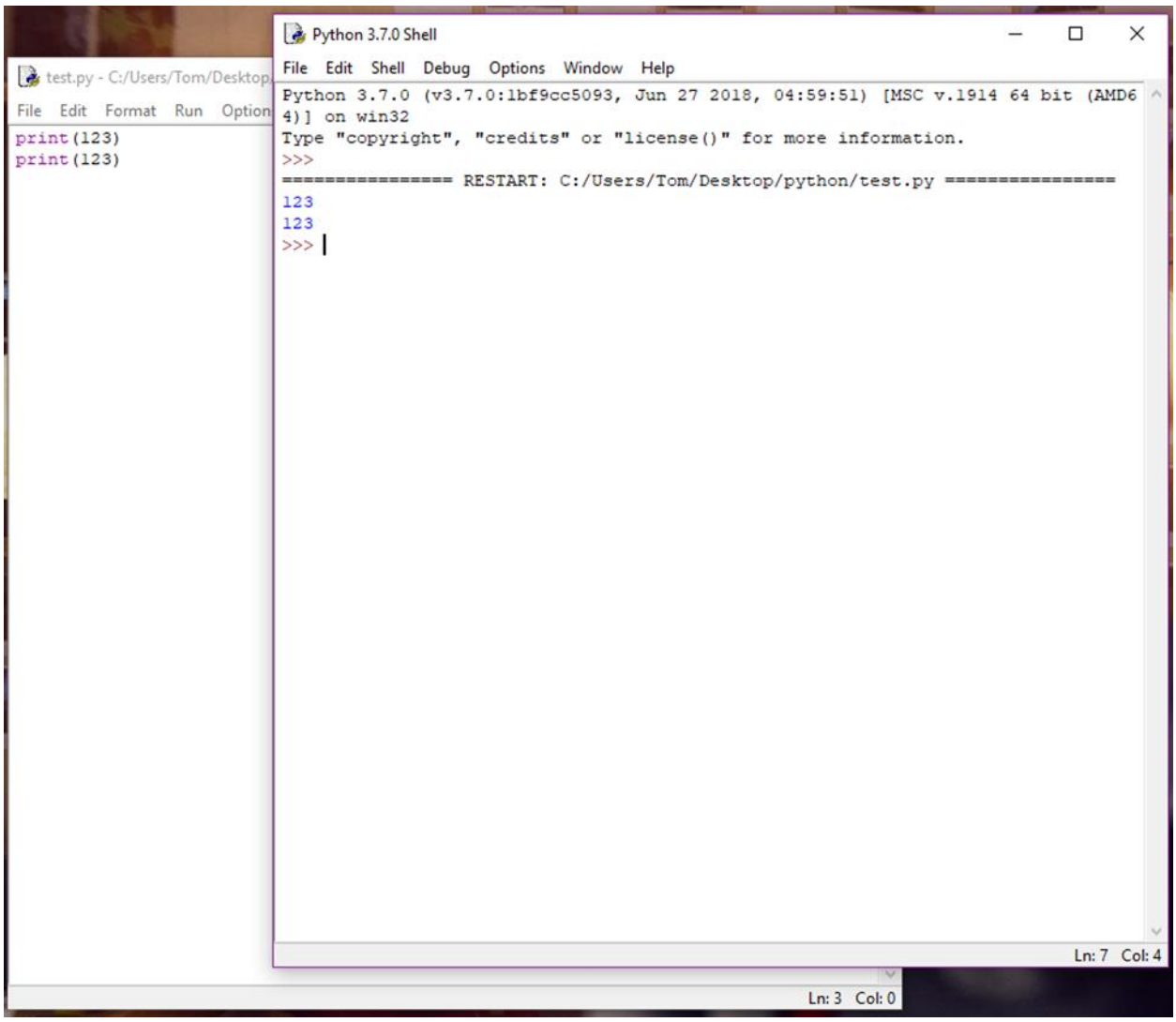


Например запишем команду

print(123)

print(123)

и нажмем F5, и как видим, IDLE запустил этот файл.



Код программы пишется в текстовом файле, потом сохраняется с расширением *.py.

Подготовить скрипты можно в той же среде IDLE. Чтобы открыть редактор кода, нужно нажать File > New File. Открывается окошко, в котором вы можете писать абсолютно любой код, любой длины, в любом количестве. Затем желательно сразу сохранить файл в расширении *.py. Если набирать код, не сохранив файл в начале, то синтаксис не будет подсвечиваться. После того как вписали весь нужный код, сохраните файл ещё раз, чтобы обновить сохранение. Теперь можно запустить скрипт, выполнив команду меню Run > Run Module или нажать клавишу F5. Если вы не сохранили, то всплывающее окошко предложит вам сначала сохраниться. После этого в первом окне (где «работает» интерпретатор) IDLE автоматически запускает файл. Там где мы писали код, можно нажать F5 и файл снова запустится.

Скрипты можно также писать в любом текстовом редакторе. Также существуют специальные программы для разработки, которые предоставляют дополнительные возможности и удобства.

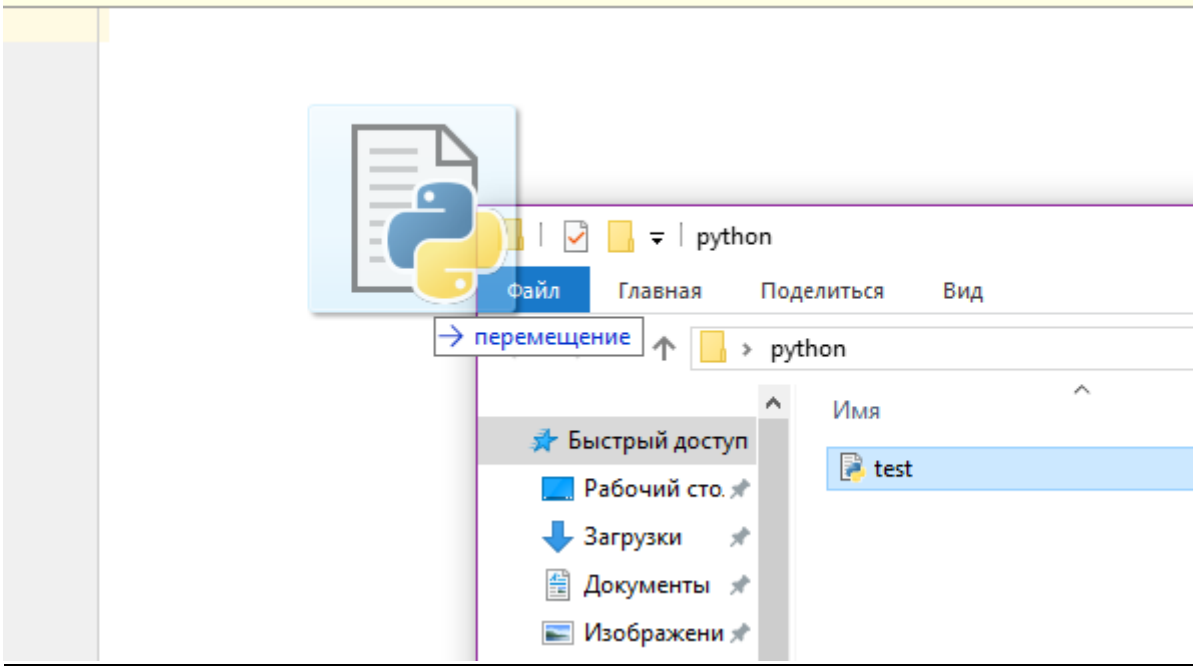
Запускать подготовленные файлы можно не только в IDLE, но и в консоли с помощью команды python адрес/имя_файла.

Также, существует возможность настроить выполнение скриптов с помощью двойного клика по файлу (в Windows данная возможность присутствует изначально).

Можно использовать какой-нибудь внешний редактор, и запускать код через консоль.

Редакторов для написания кода на python может быть много, тут каждый может использовать то, что удобнее. Можно использовать любой текстовый редактор.

Для того чтобы открыть файл во внешнем редакторе, его нужно просто перетянуть

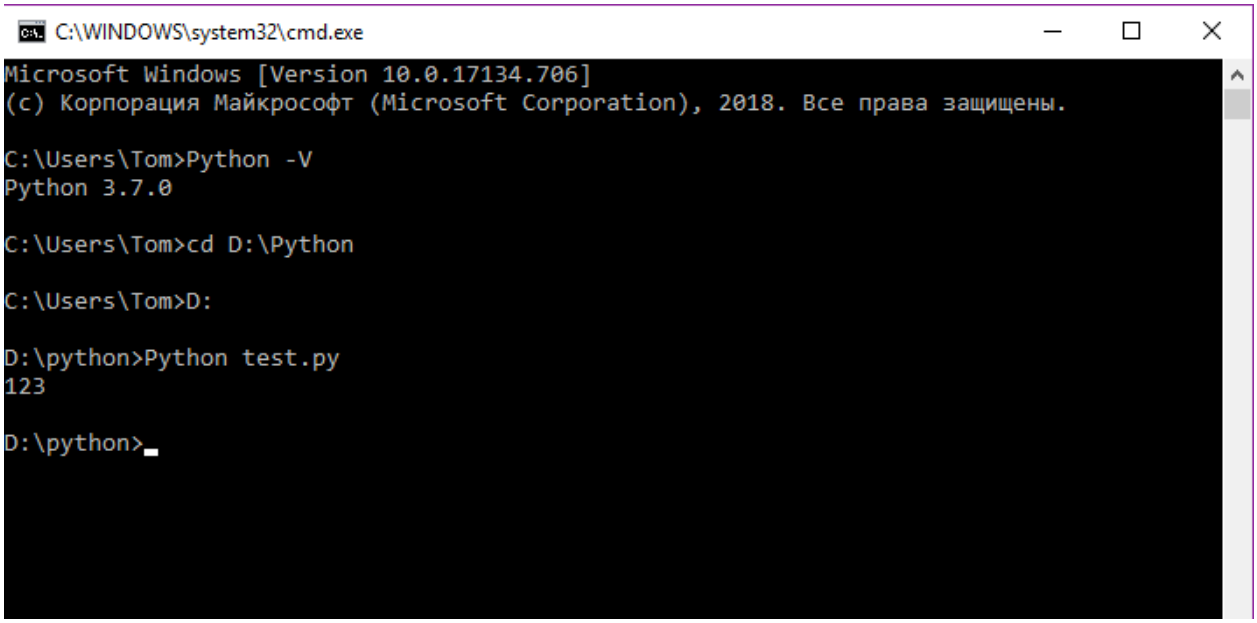


Для того чтобы запустить код написанный во внешнем редакторе кода, нам понадобится снова командная строка. Из командной строки нужно перейти в ту директорию в которой находится наш скрипт.

1. Мы сохранили файл на диске D в папке Python, поэтому вбиваем команду cd D:\Python нажимаем ввод

2. Затем меняем букву диска на D, в следующей строке вводим D: нажимаем ввод.

3. В следующей строке вводим команду Python test.py, где «test.py» - название нашего файла, нажимаете ввод.



Теперь мы видим что наш код выполнился.

Итак, мы познакомились с двумя вариантами как начать писать и как запускать код python.

Закрепление изученного материала – 17 мин.

Учитель задает вопросы:

1) Назовите эти 2 варианта
2) Как сохранить код в файл?

Конспект урока № 2.

Тема урока: Типы данных. Определение переменной. Ввод и вывод данных. Среда Geany Geany и PyScripter для написания программ на языке Python .

Тип урока: урок усвоения новых знаний.

Цель урока: познакомить учащихся с типами данных, вводом и выводом данных, с переменными, со средой Geany Geany и PyScripter для написания программ на языке Python

Задачи урока:

Образовательная: сформировать представление о языке Python, научиться писать простой код на ЯП Python в среде разработки.

Развивающая: развивать у учащихся логическое и алгоритмическое мышление, навыки мыслительной деятельности, включая каждого учащегося в учебно-познавательный процесс и создавая условия для работы каждого в индивидуальном психологическом ритме;

Воспитательная: воспитывать у учащихся самостоятельность, активность, интерес к предмету.

Этапы урока:

1) Организационный момент – 1 мин.

2) Изучение нового материала – 25 мин.

3) Рефлексия – 2 мин.

4) Закрепление изученного материала – 17 мин.

Оборудование:

- Компьютер, Python (версия не ниже 3.5), модули Tkinter и NumPy, среды разработки на Python: IDLE, Eric или Geany, а также какие-либо эмуляторы терминалов _ xterm, rxvt, проектор.

СТРУКТУРА И ХОД УРОКА

1. Организационный момент 1 мин.

Учитель: *Приветствует класс, проверяет присутствующих.*

Здравствуйте, ребята. Сегодня мы с вами познакомимся с типами данных, вводом и выводом данных, с переменными, со средой Geany Geany и PyScripter для написания программ на языке Python

2. Изучение нового материала – 25 мин.

Учитель: *Демонстрирует презентацию и рассказывает новый материал*

Данные и их типы.

Для начала, чтобы познакомиться с языком программирования Python мы рассмотрим 5 типов данных:

1. целые числа (integer, сокращённо int) – положительные и отрицательные целые числа, включая 0 (например, 4, 687, -45, 0);

2. числа с плавающей точкой (float, сокращённо str) – дробные (вещественные) числа (например, 1.45, -3.789654, 0.00453).

3. строки (string) – набор символов, заключенных в кавычки (например, «Велосипед», «Как зовут вашего кота?», 'ddfKkc', '12345'). Кавычки могут быть одинарными или двойными. В строке могут быть любые символы на любом языке – слово, предложение, набор чисел и т.д.

4. bool – имеет два значения true или false

4. Словари – неупорядоченные коллекции произвольных объектов с доступом по ключу. Их иногда ещё называют ассоциативными массивами или хеш-таблицами.

5. Списки – упорядоченные изменяемые коллекции объектов произвольных типов (почти как массив, но типы могут отличаться).

Ввод и вывод данных

Ввод данных с клавиатуры в программу (начиная с версии Python 3.0) осуществляется с помощью функции input(). Когда данная функция выполняется, то поток выполнения программы останавливается в ожидании данных, которые пользователь должен ввести с помощью клавиатуры. После ввода данных и нажатия клавиши Enter, функция input() завершает свое выполнение и возвращает результат, который представляет собой строку символов, введенных пользователем.

Функция вывода называется print(). В скобках указываются так называемые аргументы функции. Может выводить и числа и строки и что угодно. Допустим, если указать название переменной, то выведет на экран то, что хранит в себе переменная.

```
>>> input()
1234
'1234'
>>> input()
Hello World!
'Hello World!'
>>>
```

Рис. 1. Ввод данных с клавиатуры.

Функция input() может принимать необязательный аргумент-приглашение строкового типа; при выполнении функции сообщение будет появляться на экране и информировать человека о запрашиваемых данных.

```
>>> input("Введите номер карты: ")
Введите номер карты: 98765
'98765'
>>> input('Input your name: ')
Input your name: Sasha
'Sasha'
>>>
```

Рис. 2. Ввод данных с клавиатуры.

Из примеров видно, что данные возвращаются в виде строки, даже если было введено число. В более ранних версиях Python были две встроенные функции, позволяющие получать данные с клавиатуры: raw_input(), возвращающая в программу строку и input(), возвращающая число. Начиная с версии Python 3.0, если требуется получить число, то результат выполнения функции input() изменяют с помощью функций int() или float().

```
>>> input('Введите число: ')
Введите число: 10
'10'
>>> int(input('Введите число: '))
Введите число: 10
10
>>> float(input('Введите число: '))
Введите число: 10
10.0
>>>
```

Рис. 4. Применение функций int и float при вводе данных с клавиатуры.

Результат, возвращаемый функцией input(), присваивают переменной для дальнейшего использования в программе.

```
>>> userName = input('What is your name? ')
What is your name? Masha
>>> exp = input('3*34 = ')
3*34 = 102
>>> exp = int(exp) + 21
>>> userName
'Masha'
>>> exp
123
>>>
```

Рис. 3. Присвоение переменной для дальнейшего использования в программе.

Операции.

Базовые операции: +, -, *, /, **(возведение в степень), %(деление по модулю), унарный минус, округление, Пи…

Операция – это выполнение каких-нибудь действий над данными (операндами). Для выполнения конкретных действий требуются специальные инструменты – операторы.

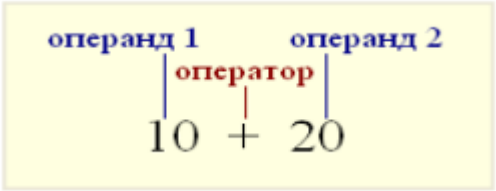


Рис. 4. Пример оператора сложение.

10+20, где «10» у нас операнд 1, «+» оператор, «20» – операнд 2

Например, запрограммировано, что символ «+» по отношению к числам выполняет операцию сложения, а по отношению к строкам - конкатенацию (соединение). Два знака «*» возводят первое число в степень второго.

программирование python лабораторный язык

Выражение	Результат выполнения
34.907 + 320.65	355.55699999999996
“Hi, ” + “world :)”	'Hi, world :)'
“Hi, ” * 10	'Hi, Hi, Hi, Hi, Hi, Hi, Hi, Hi, Hi, Hi, '

Рис. 5. Пример сложения.

Переменные.

Все данные хранятся в ячейках памяти компьютера. Когда мы вводим число оно, помещается в память. Чтобы создать переменную, мы должны ее как нибудь назвать, просто написав слово в строке, например sq, потом написать равно, и указать например цифру 4, теперь мы видим, что переменная sq хранит в себе число 5. Переменная, это ячейка в которой храним данные на время исполнения нашего кода, а потом этими данными можем как то манипулировать, изменять и удалять.

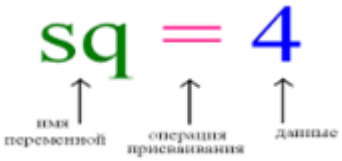


Рис. 6. Задание переменной.

Связывание данных и имени (переменной) в языке программирования Python происходит с помощью знака «=» (операция присваивания). Например, запись sq =4 означает,

что объект (данные) в определенной области памяти связаны с именем sq и обращаться к ним теперь следует по этому имени.

Имена переменных могут быть любыми. Однако есть несколько общих правил их написания:

1. Желательно давать переменным имена, несущие в себе смысловое значение.

2. Имя переменной не должно совпадать с командами языка (зарезервированными ключевыми словами).

3. Имя переменной должно начинаться с буквы или символа подчеркивания (_). Чтобы узнать значение, на которое ссылается переменная, находясь в режиме интерпретатора, достаточно ее просто вызвать (написать имя и нажать Enter). С цифры переменные начинаться не могут, а также с других спецсимволов, в таком случае мы получим ошибку.

4. Нельзя в названии переменной ставить пробел.

Задание: попробуйте создать переменную

1) 5test
2) test 5

Что получилось в данных случаях при выводе на экране?

Среда Geany для написания программ на языке Python

Geany – это свободная среда разработки программного обеспечения для UNIX–подобных операционных систем и Windows, а также для операционных систем, имеющих библиотеку GTK2. Geany использует библиотеку GTK2. Geany распространяется согласно GNU General Public License.

Geany не включает в свой состав компилятор. Вместо этого используется GNU Compiler Collection (или любой другой компилятор) для создания исполняемого кода.

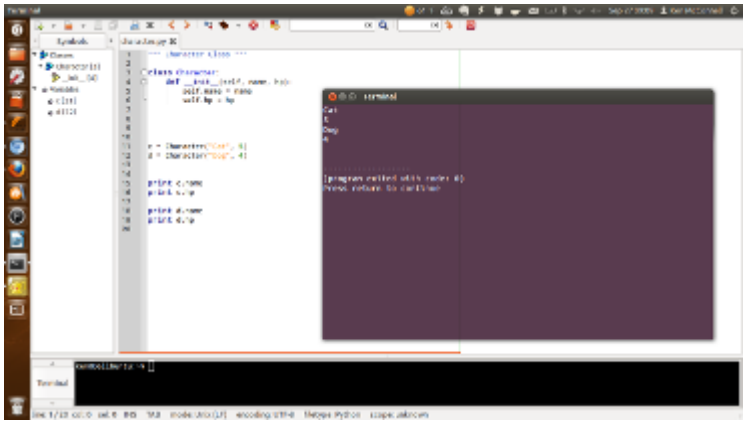


Рис. 7. Интерфейс программы Geany

Особенности:

– Подсветка исходного кода с учетом синтаксиса используемого языка программирования (язык определяется автоматически);

– Автозавершение;

– Автоматическая подстановка закрывающих тегов HTML/XML;

– Простой менеджер проектов;

– Поддержка плагинов.

PyScripter

PyScripter является редактором исходного кода, специально разработаный для языка программирования Python. PyScripter также предоставляет возможность редактирования кода, написанного на C / C + +, HTML, CSS, JavaScript, PHP, XML и т.д., но его основной целью является предоставление разработчикам решений для написания и отладки кода Python. Программа имеет открытый исходный код и работает с 32-х и 64-х битными системами. Его графический пользовательский интерфейс можно настроить по-разному для того, чтобы обеспечить максимальную эффективность. Пользователи получают возможность расположить панели инструментов в главном окне, как они хотят. PyScripter 2.5.3 поставляется с большим количеством улучшений. Тем не менее, для того, чтобы иметь возможность работать с PyScripter, вам также необходимо установить Python в вашей системе. PyScripter является мощной средой для редактирования и отладки кода Python. Он будет высоко оценен Python программистами.

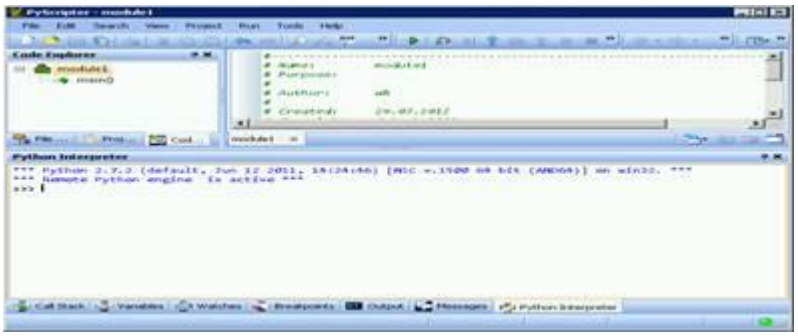


Рис. 8. Интерфейс программы PyScripter

3.Практическая часть.

Инструктаж перед выполнением практической работы по ТБ.

- В процессе выполнения практического задания учащиеся должны показать:
- Умение грамотно формулировать задачи задания.
- Умение грамотно интерпретировать результаты поставленных задач и применять эти результаты в практической деятельности.
- Умение анализировать информацию и делать выводы.

Ученики самостоятельно выполняют задания с помощью учителя.

1. Переменной var_int присвойте значение 10, var_float – значение 8.4, var_str – «No».

2. Измените значение, хранимое в переменной var_int, увеличив его в 3.5 раза, результат свяжите с переменной big_int.

3. Измените значение, хранимое в переменной var_float, уменьшив его на единицу, результат свяжите с той же переменной.

4. Разделите var_int на var_float, а затем big_int на var_float. Результат данных выражений не привязывайте ни к каким переменным.

5. Измените значение переменной var_str на «NoNoYesYesYes». При формировании нового значения используйте операции конкатенации (+) и повторения строки (*).

6. Выведите значения всех переменных.

Закрепление изученного материала – 17 мин.

Учитель задает вопросы:

1) Какие типы данных сегодня изучили?

2) Как осуществлять ввод и вывод данных?

3) Как задать переменную?

Конспект урока № 3.

Тема урока: Решение вычислительных задач.

Тип урока: урок усвоения новых знаний.

Цель урока: научить учащихся самостоятельно решать вычислительные задачи.

Задачи урока:

Образовательная: научить решать элементарные задачи в Python

Развивающая: развивать у учащихся логическое и алгоритмическое мышление, навыки мыслительной деятельности, включая каждого учащегося в учебно-познавательный процесс и создавая условия для работы каждого в индивидуальном психологическом ритме;

Воспитательная: воспитывать у учащихся самостоятельность, активность, интерес к предмету.

Этапы урока:

1) Организационный момент – 1 мин.

2) Повторение пройденного материала – 10 мин.

3) Рефлексия – 2 мин.

4) Самостоятельная работа – 22 мин.

Оборудование:

- Компьютер, Python (версия не ниже 3.5), модули Tkinter и NumPy, среды разработки на Python: IDLE, Eric или Geany, а также какие-либо эмуляторы терминалов _ xterm, rxvt, проектор.

СТРУКТУРА И ХОД УРОКА

1. Организационный момент 1 мин.

Учитель: *Приветствует класс, проверяет присутствующих.*

Здравствуйте, ребята. Сегодня мы с вами будем учиться самостоятельно решать вычислительные задачи, и применять полученные знания на практике.

Давайте повторим пройденный материал на прошлом уроке

1. Какие типы данных вы знаете?

2. Как осуществлять ввод и вывод данных?

3. Как задать переменную?

Дальше ученики отвечают, и учитель выдает задания для выполнения на компьютере.

Задание 1. Определить объем цилиндра.

Пример выполнения задания:

r=input(‘введите радиус’)

h= input(‘введите высоту’)

pi=3,14

v=pi*r^2*h

print(v)

```
r = int(input('введите радиус'))
h = int(input('введите высоту'))
pi = 3.14
v = pi * (r ^ 2) * h
print(v)
```

Рис. 1. Нахождение объема цилиндра

Задание 2. Извлеките кубический корень из суммы двух чисел вводимых с клавиатуры.

Пример выполнения задания:

x=int(input('введите первое число'))

y= int(input('введите введите второе число'))

p=(x+y)**(1/3)

print(int(p))

```
x=int(input('введите первое число'))
y= int(input('введите введите второе число'))
p=(x+y)**(1/3)
print(int(p))
```

Рис. 2. Извлечение кубического корня

Ученики выполняют, обращаются за помощью к учителю, при выполнении показывают результат.