УДК 004

Муковнина А.А.

студент

Научный руководитель: Быкова К.И., к.ф.-м.н

Воронежский государственный педагогический университет

РАЗВИТИЕ АЛГОРИТМИЧЕСКОГО МЫШЛЕНИЯ НА УРОКАХ ИНФОРМАТИКИ С ПОМОЩЬЮ ИЗУЧЕНИЯ ЯЗЫКА PYTHO

Аннотация: в статье рассматриваются методические рекомендации по развитию алгоритмического мышления посредством языка Python. Делается акцент на некоторых особенностях данного языка. Даются практические рекомендации по проведению уроков для учителей информатики.

Ключевые слова: программирование, алгоритмическое мышление, алгоритм.

Mukovnina A.A.

student

Scientific supervisor: K.I. Bykova, Ph.D.

Voronezh State Pedagogical Universit

DEVELOPING ALGORITHMIC THINKING IN COMPUTER SCIENCE LESSONS BY LEARNING PYTHON

Abstract: the article discusses methodological recommendations for the development of algorithmic thinking through the Python language. The emphasis is on some of the features of this language. Practical recommendations on conducting lessons for computer science teachers are given.

Keywords: programming, algorithmic thinking, algorithm.

Тенденции развития современного общества предъявляют совершенно новые требования к умениям и навыкам учащихся. Если в 80-ых годах компьютер был чем-то недосягаемым для обычного школьника, то сегодня каждый учащийся свободно может управляться с ним. Чтобы в будущем ученик умел не только получать знания, но и мог решать разнообразные практические и теоретические задачи, был способен самостоятельно добывать нужные ему знания с использованием современной техники, умел быстро и правильно решать возникшие в его жизни задачи и проблемы, он должен научиться думать.

Если мы обладаем желание вырастить таких людей, то необходимо формировать у учащихся разнообразные методы мышления, общие способы подхода к любой задаче и проблеме. Одним из наиболее необходимых типов мышлений в наше время является алгоритмическое мышление, которое в большинстве своем формируется на уроках информатики, математики, а также физики. Без данного вида мышления не обходится ни один процесс в реальности.

Умение составлять и решать задачи требует специального навыка – алгоритмического мышления, который вырабатывается и совершенствуется на протяжении жизни человека.

Алгоритмическое мышление можно понимать, как систему мыслительных приемов, направленных на решение задач. Тут скрыты две стороны понимания. Первая, определить чужой алгоритм. Вторая, построить свой. Если при решении какой-либо задачи вам приходится взаимодействовать с чем-либо, придется понимать, как оно устроено.

Только после понимания внутреннего строения можно встраивать свой алгоритм [3].

Алгоритмическое мышление отличается формальностью, логичностью, ясностью. С помощью него можно решать любую задачу, выполняя определенный алгоритм, построенный для решения. Именно такой вид мышления способствует успешному изучению программирования.

Знакомясь с современными методиками преподавания информатики, совершенно точно можно прийти к выводу, что курс программирования является одним из основных при изучении данного предмета [2].

К сожалению, в данное время наблюдается крайне низкий уровень подготовки школьников по вопросам программирования и алгоритмизации. После изменений, внесенных во ФГОС в 2022 году, мы видим, что изучение программирования в большинстве своем предполагается на углубленном уровне. Базовый уровень предполагает знакомство с разделом алгоритмизации в 8-ом классе, в то время как на углубленном уровне данные вопросы рассматриваются в 7-ом. Если говорить о программировании, то базовый уровень не предусматривает рассмотрение вопросов, связанных с циклическими алгоритмическими конструкциями. Данной темы касаются лишь на углубленном уровне.

С одной стороны, это правильно, теперь мы имеет четкое разграничение изучаемых тем на базовом и углубленном уровнях, но с другой стороны, вопросы, которые на базовом уровне не рассматриваются помогают наиболее точно разобраться в вопросах алгоритмизации.

Что касается языка программирования с помощью, которого в полной мере можно формировать алгоритмическое мышление, то на сегодняшний день предпочтение отдается Python. Его преимущества по сравнению с другими языками отмечают большинство учащихся. Он достаточно прост в усвоении, лаконичен, имеет легкий и понятный синтаксис, по сравнению с любым другим языком, обладает обширной сферой использования, динамичной типизацией, большим количеством библиотек и широко востребован на сегодняшнем рынке.

Язык Python – это инструмент для создания программ самого разнообразного назначения, который доступен даже для новичков. Как показывает практика, даже те, кто совершенно не знаком с программированием прибегают к знакомству именно посредством данного языка.

Стоит понимать, что, начиная раздел программирования, обучающиеся уже имеют представление об алгоритмизации, поэтому целесообразно во время проведения уроков проводить аналогию между основами алгоритмизации и начальными сведениями программирования.

Прежде чем начинать программирование на языке Python необходимо определиться со средой, в которой будут создаваться ваши программы. Мы рекомендует использовать Python IDLE. Это свободно распространяемое ПО, имеющее достаточно понятный интерфейс.

После установки ПО можно перейти к работе. Начиная работу с языком Python необходимо познакомить учащихся с типами данных, которые используются в языке. Перечислим основные типы данных в

языке Python: int – целые значения; float – вещественные значения (числа с дробной частью); bool – логические значения, true (истина, «да») или false (ложь, «нет»); str – символ или символьная строка, то есть цепочка символов. Следом перейти к арифметическим операциям и выражениям. После того, как эти вопросы будут разобраны, можно перейти к использованию основных алгоритмических конструкций [1].

Наиболее простым типом алгоритмической конструкции является конструкция следования, согласно которой действия выполняются последовательно друг за другом. В качестве примера данной конструкции можно рассмотреть следующие задачи: сложение нескольких чисел, поиск среднего арифметического, периметров и площадей различных фигур и др. На Python решение данных задач предполагает использование всего двух функций: input( ), print ( ). Первая предполагает ввод информации, вторая ее вывод на экран. При написании программ данного вида стоит отметить, что данная среда воспринимает введенные числа, как символы, поэтому, если вы ввели два числа, первое – 2, второе – 3, на выходе вы получите 23, а не сумму равную 6. Для того, чтобы корректно работать с данными необходимо указывать их тип. Например вместо команды a = input ( ), писать a =int (input ( )), давая понять вашей программе, что вы работаете с целым числом [1, с. 8-9].

Следующая конструкция – ветвление. Данная конструкция реализуется с помощью условного оператора if. Также возможно использование if и else. В Python важную роль играют сдвиги операторов относительно левой границы (отступы). Обратите внимание, что слова if и else начинаются на одном уровне, а все

команды внутренних блоков сдвинуты относительно этого уровня вправо на одно и то же расстояние. Для сдвига используют символы табуляции: одно нажатие на клавишу Tab или четыре пробела. Если необходимо ввести несколько альтернативных условий, то можно использовать дополнительные блоки elif (сокращенное от else – if), после которого идет блок инструкций [1]. В качестве примера данной конструкции можно рассмотреть алгоритм, поиска корней квадратного уравнения или любые алгоритмы, предполагающие условие.

Последняя алгоритмическая конструкция – повторения, в рамках которой организуется работа с циклами. Циклы являются такой же важной частью программирования, как условные операторы. С их помощью можно организовать повторение некоторых частей кода. В Python для записи циклов используются 2 вида команд: while и for. While переводится с английского как «пока», то есть цикл (блок команд) выполняется до тех пор, пока не выполнится заданное условие. Для этого в начале очередного шага цикла выполняется проверка условия. Поэтому оно называется циклом с предусловием. Цикл for повторяет команды необходимое количество раз. Данная команда позволяет сделать программу компактнее. В рамках изучения циклов можно рассмотреть задачи на определение количества цифр в числе, суммы цифр числа и т.д.

После рассмотрения основных алгоритмических конструкций рекомендуется познакомить учащихся со сложными условиями.

В рамках изучения раздела программирования можно параллельно показывать обучающимся насколько компактнее выглядит программа, написанная на языке Python.

Стоит отметить, что алгоритмический подход применим не только в компьютерных науках, но и в любых других. Алгоритмическое мышление необходимо развивать, чтобы понимать внутреннее устройство чего-либо. Для решения любой задачи приходится с чем-то взаимодействовать, и для построения своего алгоритма необходимо понимать строение основной системы. Но все это возможно лишь тогда, когда у человека есть желание думать [3].

Использованные источники:

1. Программирование на языке Python для школьников: Учебное пособие по изучению языка программировани Python / Л. Самыкбаева, А. Беляев, А. Палитаев, И. Ташиев, С.Маматов – Фонд Сорос-Кыргызстан, 2019 – 84 с.

2. Стась А.Н., Прусских О.Н. Формирование алгоритмического мышления в процессе обучения теории графов//Вестник ТГПУ. - 2012. - №2. - С.166

3. Чебурина О.В. Формирование алгоритмического мышления в обучении программированию игр//Наука и перспективы. - 2017. - №2. - С. 2-3.