

О языках программирования и их изучении

Наумцев Александр

from 10 October to 20 October

1 Первое знакомство с языком программирования

Со временем, помимо математики захотелось открыть для себя новый инструмент решения задач - программирование. Учителей и времени особо не было, поэтому по-началу у брата-ровесника спрашивал базовые вещи на уровне синтаксиса C++ (циклы, условия, вывод, рекурсия). В какой-то момент появился отбор в сирius с методочками по контейнерами, так я освоил (одно из самых важных в олимп проге, да и вообще) контейнеры в C++. После некоторого времени решения (местные контесты, олимпиады, командные олимпиады) задач случилось, что по региону попал в Сириус и в ЛКШ, что дало понимание, того что можно решать большинство практических задач (любые какие появятся в голове и это круто). Опыт олимпиадного программирования не такой как математический, он очень крутой, но не стоит уделять этому всё время, я считаю лучше делить время 65% математика и 35 % программирование.

2 Дальнейшее обучение

Потихоньку, решая чётко поставленные задачи, мне стало хотется делать, что-то для жизни более наглядное - решением этого пришёл язык Python. После того как я увидел, что брат научился делать в Яндекс лицее (БД, сайты, работа с различными файлами, музыкой, картинками, навык в Алисе оценённый премией от Яндекса, победа на выпускном Яндексе) мне стало это жутко интересно. Изучения синтаксиса Python после C++ показалось немного примитивным, но это пропало, когда увидел как можно навешивать большие конструкции. Python дал понимание, того что можно сделать практически всё, только нужно понять как этим пользоваться. Так я участвовал в AI CUP, писал telegram bot's и многое другое. Здесь я понял, как важно и очень классно изучать язык, у которого ты можешь видеть практическую пользу.

3 Выбор следующего языка

Понятно, что для каждого задания, найдётся язык, который подойдёт лучше для него, но хочется научиться делать разнообразные вещи на хотябы на одном языке, Python подходит под это описание. Поэтому я не вижу большого смысла изучения нового языка, только если это будет нацелено на привыкание к чему-то новому (синтаксис, новые идеи написания кода). Я бы поизучал язык Java, что бы привыкнуть к нему, так как на нём можно выполнять задачи, которые делать на C++ и Python не эффективно.

4 План по изучению языка Python

Рассказ о изучения C++ я уже рассказал, но этот план на изучение олимпиадного программирования. Про Python говорить легче, в нём легче и быстрее выходить на стадию разработки всяких классных практичных вещей.

4.1 Изучение синтаксиса

В любом языке, чтобы освоить синтаксис самым полезным способом будет - практика. Ищите либо курс на степике, либо просто сайт с синтаксом и идёте на [codeforces/acmp/informatics](https://codeforces.com/acmp/informatics) и прорешиваете задачу за задачей. Когда вы чувствуете, что вы освоили циклы, условия и рекурсию (функции) переходим дальше.

4.1.1 Контейнеры

Дальше стоит изучить контейнеры: map, set, vector (C++) В разных языках они называются и делают общую вещь, но в некоторых языках они могут выполнять гораздо больше, чем от них требуется.

4.1.2 Полезные функции

В каждом языке есть кучу полезных функций, которые помогают быстро написать программу и сделать код более читабельным. Освоили? Значит возвращаемся на сайты с контестами и пытаемся решать уже более сложные задачи с какими-то алгоритмами.

4.2 Разработка

Осваиваем работу с памятью, с классами и пытаемся работать с файлами, а. А дальше уже, начинаем делать, что вам нравится. Делаем БД, создаём сайты, пишем ботов. Как мы выяснили раньше, главное разбраться.

4.3 Командная работа

После получения какого-то опыта работающего кода, стоит начинать работы в команде, ведь написать код можно по-разному, а вот эффективно и красиво уже задача по-тяжелей в этом может помочь командная работа и чей-то опыт.

4.4 Python глазами C++

Эти два языка по-раздельности могут делать много чего. Python может вдохновлять своей простотой и возможностями, но иногда нужно сделать часть в проекте, которая должна работать быстро, что Python не может дать или может, но нужно заморочиться. Хочется, что была библиотека контейнеров C++ в обёртке Python, почему-то такого нету или не так популярна. Я думаю это можно реализовать не потеряв читаемость кода и стиль Python.

Почему хочется дополнить Python, а не C++? Python простой для освоения и реализации многих задач, что не скажешь про C++. В целом он отличный выбор программисту и его в большинстве, нужно оставить как и есть. Просто хотелось бы в некоторых местах 'заглядывать под копот' и писать код на чуть низком уровне.