ТУБ №1:
Этапы постановки и решения задачи на компьютере

С чего начинается написание компьютерной программы? Почему одни программы мы любим, а другие стараемся обходить стороной? Почему у каждого человека свои предпочтения при выборе программ? Как создать программу, которая понравится всем? Почему это невозможно и почему это не повод бросить всё и уйти в маляры?

Давайте представим себе, что Вы решили открыть фирму, которая занимается производством автомобилей под заказ. Офисное помещение уже обставлено мебелью, распечатаны футболки с логотипом компании, на маленькой, но уютной кухоньке жужжит новенькая кофе-машина, а на всех Ваших страницах в соцсетях красуется Ваша светящаяся от счастья харизма на фоне начищенной до блеска новенькой яхты — в общем, Вы уже по всем канонам успешный бизнесмен.

И вот приходит Ваш первый заказчик и говорит:

— Нужна машина, заказывать будем у вас. Понимаю, что будет недёшево, но это не проблема, заплатим, сколько потребуется.

Облака, до этого закрывавшие собой половину неба, расступаются и дают дорогу солнечному свету, мир вокруг начинает играть яркими красками, откуда-то с небес доносится едва уловимая мелодия, в которой в унисон слились звуки счастья и вечной благодати, воображение рисует заманчивые перспективы и острова в Тихом океане, которые вот-вот станут принадлежать Вам.

«Это ещё надо сначала с поставленной задачей справиться», — спускает Вас с небес на землю голос противного ворчуна, имя которому Здравый Смысл.

Сделать всё правильно непросто. Вот, например, приступаете Вы к работе, создаёте суперэкономичный автомобиль, который заставить глотать пыль любую гоночную машину. Делаете его небольшим, манёвренным, но очень вместительным и комфортным. В общем, автомобиль-сказка! Моете до блеска, повязываете ленточку, а потом приходит заказчик и говорит:

— Ой, а я разве не предупредил? Это ж для армии. Нам надо с неё ракеты класса «земля-воздух» запускать.

Проходит неделя, которую Вы проводите, прикручивая к своей лапочке-машине ракетные установки, попутно укрепляя крышу, чтобы при выстреле залпом наводчик не падал на голову водителю. Приходит заказчик, Вы с гордостью представляете ему полученный результат и слышите:

— Погодите-погодите, а она, если её на болоте использовать, не утонет?

Проходит несколько лет, Ваша машина всё больше похожа на космический корабль инопланетной цивилизации, в ней появляются Bluetooth, 42-мегапиксельная барокамера, лазерный телескоп, встроенный аппарат МРТ, кнопка вызова метро и открывашка для консервных банок, а завершение проекта так и не предвидится…

Критично настроенный читатель скажет, что открывашка — это слишком надуманно, и с этим сложно не согласиться. Но подобные ситуации действительно случаются в программировании, причём нередко. Есть статистика [1], которая показывает, что только около трети IT-проектов завершается успешно, т.е. без непредвиденных расходов, в полном объёме и в срок. Получается, успешно справиться с разработкой программы сложнее, чем угадать результат подбрасывания монетки. Это при том, что на разработку программы, в отличие от монетки, можно повлиять!

Давайте проанализируем нашу гипотетическую ситуацию. Какая ошибка была допущена с самого начала?

Совершенно очевидно, что перед тем, как приступить к разработке, мы не удосужились уточнить детали задачи. Полученный результат казался нам идеальным, но не соответствовал ожиданиям заказчика.

Попробуем перенести это наблюдение на программирование. Предположим, заказчик сформулировал нашу задачу так: «Нужна программа для журналистов, которая на фотографиях будет находить и обводить лица известных людей». Давайте попробуем понять, что и в какой последовательности нужно делать, чтобы повысить наши шансы на успех? Другими словами, давайте попробуем выделить этапы постановки и решения задачи на компьютере.

1. Чёткая формулировка задачи, выделение исходных данных, результатов и формы их представления.

Будет ли полезна нашему заказчику программа, которая обведёт на фотографии лицо куклы или манекена? Какие люди считаются известными? Обвести лицо — это как? По контуру лица? Прямоугольником? Овалом? Что делать, если лиц на фотографии нет?

Ответы на эти вопросы могут сильно повлиять на сложность задачи, а значит, и объём работ, иногда после такого уточнения задача из очень простой превращается в сложную или нерешаемую. Или наоборот.

Особое внимание здесь следует уделить тому, какие данные программа сможет получить от пользователя и что должна будет вернуть в ответ. Если известные люди — это те, информацию о ком пользователь задал сам, это один случай. Если же программа сама должна будет составлять и обновлять список знаменитостей, это совсем другая история. А может быть, заказчик имел в виду, что пользователь сам будет обводить лица и нужно просто сделать этот процесс максимально комфортным по сравнению с существующими графическими редакторами?

Форма представления исходных данных и результатов тоже оказывает влияние на решение проблемы.

Откуда программа берёт фотографии на обработку? Загружает с подключённого к компьютеру цифрового фотоаппарата? Делает стоп-кадры с веб-камеры? Пользователь вручную выбирает файл с изображением? В каком из популярных форматов это изображение? JPEG? PNG? Или пользователь выбирает сразу целую папку с фотографиями и программа должна сама найти в ней фотографии? Или эта папка заранее задана и программе нужно отслеживать появление в ней новых файлов?

Результат нужно сохранить в файл? Какого формата? Того же, что и исходное изображение? Или заранее заданного? Нужно ли приводить все изображения-результаты к одному и тому же размеру? А может быть, полученная фотография должна быть сразу же размещена в определённом фотоальбоме в какой-то из социальных сетей? Или вставлена в видеотрансляцию, которую пользователь прямо сейчас ведёт со своего компьютера? Или нужно просто показать результат на экране?

2. Формальная (математическая) постановка задачи.

Здесь имеется в виду представление задачи в виде уравнений, соотношений и ограничений. Что именно мы будем считать лицом на фотографии? Может ли лицо на фотографии быть сильно вытянутым или нужно ограничиться нормальными пропорциями? В каком случае человек на фотографии считается достаточно известным? Сколько раз он должен быть упомянут в СМИ? Или известность определяется количеством просмотров у его канала на YouTube?

3. Выбор метода решения.

Как именно мы будет среди отдельных пикселей изображения находить группы, похожие на лица? Будем придумывать свой метод или воспользуемся уже существующими, например, нейронной сетью?

4. Разработка алгоритма решения задачи.

Теперь, когда мы определились с методами решения задачи, нужно понять, как именно эти методы будут работать вместе для достижения результата, какие могут возникнуть ситуации, требующие особой обработки и т.д.

По сути, именно на этом этапе мы впервые увидим очертания нашей будущей программы.

5. Выбор структур данных.

Здесь под структурой данных понимается то, чем с точки зрения программы будут фотографии, лица и т.д. Есть разные способы организовать хранение данных в памяти компьютера, и от выбора структуры данных во многом зависит и алгоритм их обработки, поэтому этапе 4 и 5, вообще говоря, довольно сильно связаны между собой.

6. Программирование.

Ура, наконец-то! К этому моменту мы уже разобрались, что именно и как именно будем делать, а заодно решили, какие структуры данных будем использовать. Теперь можно выбрать язык программирования, который лучше всего подойдёт для наших целей, и приступить к воплощению нашей теперь уже хорошо продуманной идеи в жизнь.

Конечно, есть программисты, которые, изучив один язык программирования, считают себя достаточно грамотными, чтобы приступить к работе. Но если всё, что у Вас есть, — это молоток, то всё вокруг будет казаться гвоздями. Поэтому-то особенно важно уметь пользоваться разными инструментами — т.е. знать разные языки программирования. Чем сильнее будут отличаться друг от друга известные Вам языки программирования, тем шире будет Ваш выбор, а значит, и Ваши возможности.

7. Тестирование и отладка программы.

Недостаточно написать программу. Человеку свойственно ошибаться и даже в сравнительно небольших программах крайне редко удаётся обойтись без ошибок. Но ведь никто не любит программы с ошибками! Поэтому приходится проверять правильность работы программы на каких-то примерах и сверять полученные результаты с ожидаемыми — тестировать программу.

Иногда при обнаружении ошибки сразу понятно, как её исправить. Но часто бывает так, что причина совсем не на поверхности. Тогда начинается самая настоящая работа для сыщика. Программа берётся под наружное наблюдение, ей снова и снова подсовывают проблемный пример и следят за тем, как она себя поведёт. Затем выясняют, в какой момент что-то пошло не так, и постепенно приближаются к разгадке. Это и есть отладка.

8. Выполнение программы на компьютере.

Вы же помните, что мы обсуждали этапы постановки и РЕШЕНИЯ задачи на компьютере? Наша задача заключалась в том, чтобы выделять лица на фотографиях. При тестировании и отладке мы могли использовать какие-то тестовые фотографии, но теперь, когда все ошибки исправлены (по крайней мере те, о которых мы знали), наступает радостный и волнительный момент запуска программы в эксплуатацию.

Здесь же, кстати, становится окончательно понятно, насколько успешно решена задача: ведь здесь на вход программе начинают подаваться именно те данные, для которых она была разработана, а значит, именно сейчас можно оценить, например, скорость работы программы и её удобство для конечного пользователя в тех условиях, в которых программа будет использоваться.

Конечно, не для всех программ эти этапы одинаковы. Если задача несложная, то некоторые их них могут становиться почти незаметными, а то и вовсе пропускаться. К тому же, с опытом программист начинает многое из того, что мы обсудили, делать «на лету» в уме, прямо по мере того, как происходит написание кода. Однако следует помнить одно важное правило:

Чем раньше тот этап, на котором допущена ошибка, тем сложнее будет её исправить.

Поэтому-то особенно важно, чтобы каждый из этапов прорабатывался достаточно хорошо. Именно от этого зависит, получится ли у нас то, чего ожидал заказчик.

Так почему же нельзя создать программу, которая понравится всем? Да потому что задачи у всех разные. По этой же причине разнятся и предпочтения. На вкус и цвет, как говорится. Так что не спешите уходить в маляры: в программировании тоже ещё много заборов некрашеных!

[1] http://foykes.com/statistika-uspeshnosti-it-proektov/