## 1　はじめに

えっと，スマホアプリを python で書くことはすごくめんどくさ (一般的ではな) くて，基本的にスマホアプリは，android なら java という言語を使って android studio というツールで，iphone なら swift という言語を使って Xcode というツールで作成します (iphone 版と android 版があるのはそういうことです)．python は GUI が弱い (画面表示が苦手) ので，そもそもアプリに向いていません (主観)．

それで，一口にプログラミングといってもだいたい下の三種類くらいに分類されて，それぞれ適切な言語がだいぶ違います (主観)．

- 1. 目に見えるもの (スマホアプリとか)
- 2. 自動化系 (めんどくさい処理を自動化)
- 3. 変態 (プログラミング言語を作るやつら)

この中で，いわゆる計算機科学的視点から見ると，1 はあまり良くありません (いいプログラミング習慣がつきにくい (これはやっていくとわかると思います))．

## 2　どれからやるべきか

ぼくは自動化方面 (数値計算とかそっちあたり) から入門したのでなんとも言えませんが，スマホアプリを作りたいってことは 1 番目を目指していると思います．スマホのようにプラットフォームに依存しない一番目として，web ページを作ってみるのがいいんじゃないでしょうか．単純な web ページにはプログラミングも何もないですが，ある程度凝ったものになると程よくプログラミング習慣をつけれると思います．一応アプリ開発を最初からゴリゴリやることはできますが，プログラミングに存在している概念 (スコープとかその辺) さえつかめば，複数のプログラミング言語を用いて開発ができるので (もちろん，プログラミング言語によって異なる”単語”は覚えなくてはいけませんが)，比較的優しいプログラミング言語から始めることをお勧めします．気持ちとしては，日本語と韓国語は表記が全く異なるけれど，文法上のルールはほとんど同じなので，単語さえ gg れば十分に読み書きできる．といったところでしょうか (わかりにくかったらごめん)

## 3　どんな言語がいいのか

プログラミング言語といっても星の数ほどありますよね．どれを選んだらいいかわかりにくいと思います．分類にもよりますが，基本的にプログラミング言語は静的型付けと動的型付けに分類できて，python は前者，c,c++ は後者に当たります．数学的な厳密さを求めるなら後者なのですが，如何せん学習が難しいので最初は前者をお勧めします．

また，python や processing 以外を windows で開発することは個人的には苦行でしかないです (設定が本当にめんどくさくてハゲそうになります)．なので virtualbox のような仮想化システムを使って windows 上で ubuntu などの linux を使って開発することをお勧めします．

## 4　環境の作り方

プログラミングをする環境の作り方 (コンパイラーやエディターの設定等) ですが，基本的には参考とする文献や web ページに書いてあると思います．先ほど，processing と python は windows で開発できることをにおわせる記述をしましたが，まさにその通りです．processing はこいつをダウンロードすれば基本なんでもできます．`https://processing.org/download/`

python はこいつをダウンロードすれば問題ありません．`https://www.anaconda.com/products/individual`

これ以外はめんどくさすぎて windows で触ったことがありません．なので環境の作り方とか知らないごめん．

## 5　学習の仕方

プログラミングの概念であったり手続きを学習したいのならただひたすら書きます．そして人が書いたコードを読みます (競技プログラミングにおいてはこれが重要です)，そうでなければ延々とコマンドを紙に書いて暗記できるようにしましょう (情報のテストは本当にクソです．手書きさせんな)．英単語とは異なり，プログラミングをする環境にはネットがあります．なのでわからなくなったらすぐに gg ります．写経もかまいませんが，写経しているコードがどのような動作をするか理解した上で写経しないとあまり意味はありません．とりあえず github に登録しましょう (n 予備校にやり方は全部書いてあります)．言ってくれればぼくの書いているコードをわたしますし，コードレビューもします．

## 6　各分野の特色と参考になるもの

それでは，一つづつ挙げていきます．

### 6.1　0 について

何も作らずにやり方のみを学習する場合．

- python について:SFC の人が書いてるっぽい https://github.com/kaityo256/python_zero
- python について:sega が書いてくれてるぜ http://techblog.sega.jp/entry/2020/04/27/100000
- c++ について:ドワンゴの中の人が書いてるっぽい https://github.com/EzoeRyou/cpp-intro
- c++ について:今日プロの人が書いてるよ https://atcoder.jp/contests/APG4b

### 6.2 1 について

スマホアプリは大変なので，web ページを作ることやそれ以外の優しい言語を用いて表現する系統をお勧めします．そんな時に便利なのが，N 予備校のプログラミング入門 web アプリです．これの 1 章 2 章を読むと web ページを作れるようになります．

https://www.nnn.ed.nico/

---

これでもいいんですが，なんか面白くないし物足りないなぁと思ったら，こういう画像を作るのはどうでしょうか？

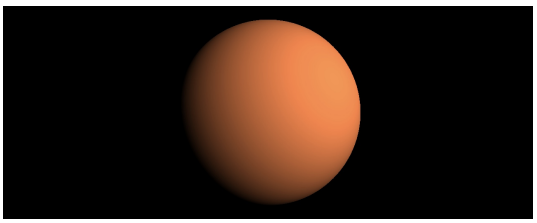

図 1　一

これは，レイトレーシング (Geforce RTX のアレ) です．これもプログラミング言語を用いて作成することができます．実際にこれはぼくが書いたプログラムによって作成された代物です．レイトレーシングにおいては，数学の知識をかなり使うのでなかなかやりがいがあると思います．また，processing という python に似た優しい言語を使うので，やりやすいと思います．

https://www.raytracing.xyz/

(対象ユーザーが小学生以上とかいう煽り)

---

さらに発展させて，数学とプログラミングを合わせる方向で行くなら競技プログラミングをお勧めします．

https://atcoder.jp/

これやると数学力が相当つく代わりに計算力が落ちるそうです．しかし，頑張れば基本的になんとかなります．

### 6.3 2 について

2 はなんらかの目標があって，それを実現するためにしぶしぶ必要になって (手動でやるのがバカバカしすぎるために) やるものばっかりです．例えば，youtube 等の動画サイトから動画を (違法に) 落としてくるツールの作成 (youtube-dl) や，点字ブロックを画像から検出するツール，月と地球の軌道を計算するツール等々．程よく簡単な目標として (休校期間中だらけないようにぼくがやっているもの)，”日本語は横書きも縦書きも可能だが，英語は横書き限定である．これは日本語と英語その他の言語における文字について，縦線と横線の割合は異なることによるものか．”という目標があるとして，各言語の縦線と横線の割合を自動で計算するプログラムの作成．とかがあります．(身近にある疑問を解決する時にこの力技を使うとすごく楽しいです．)

- 0 からつくる deeplearning:有名です．　https://www.amazon.co.jp/dp/4873117585

### 6.4 3 について

これは完全に変態向けです．いわゆるコンパイラや OS などを作る爆絶変態コースです．完成させると，相当の技術力がつきますが，途中まで進めてもかなりの知識がつくと思います．わたしの先輩にはブラウザを作ったバケモノがいます．

- コンパイラの作り方:　https://www.sigbus.info/compilerbook
- ブラウザの作り方:　https://limpet.net/mbrubeck/2014/08/08/toy-layout-engine-1.html
- os の作り方:　https://www.amazon.co.jp/dp/4839919844
- cpu のエミュレーターの作り方:は？　https://book.rvemu.app/

### 6.5 おまけ

- 群衆の叡智:Twitter を使うものとしては履修しておきたいよね．
  https://ncase.me/crowds/ja.html
- 東大の講習会:これが無料ってまじ？
  http://ong.iis.u-tokyo.ac.jp/ong-steam-stream/
- 東大のスパコン講習:読むと計算機がどうやって動いているかわかるかも．
  https://www.cc.u-tokyo.ac.jp/events/lectures/materials/

- スパコンプログラマ入門:一週間でなれるらしいぜ.
  https://github.com/qulacs/
  quantum-native-dojo
- 量子コンピュータ入門:すげえ，重ね合わせエンジニアだ.
  https://github.com/qulacs/
  quantum-native-dojo
- 漢字の成り立ちについて:全然かんけえねえけど.
  https://note.com/nkay/n/n749b4ac54acb
- やりきる能力について:ぼくは途中で捨てがちです.
  https://www.nature.com/articles/
  s42003-020-0930-4.pdf