解決済

プログラミングについて

C Perl

2016/05/29 16:28 投稿

abc_z
score 8

プログラミングが本当に苦手です。
ソースコードを見れば、何となくですがどうゆう動きがするのかは理解できるんですが、自分でソースコードを書くとなると本当に何をしたらいいのかわからずに自分に対して劣等感を感じてなりません。

プログラミングをするうえで、どのような順序でプログラミングについて勉強するとソースコードが上手く書けるようになるのでしょうか。

アドバイスお願いします。

19 回答 2 評価 720 view

0 0 0

情報の追加・修正の依頼をする(1)

Good 2 評価を下げる(0) クリップ

新着順 評価が高い順 古い順

回答（全19件）

10

どうプログラミングしたらいいのかわからないって時は、だいたい二つの問題に分けられると思うんだ。

プログラミングのやり方がわからない。

よくある入門書は「プログラムがどう動くか」は説明しても「プログラムをどう書くか」は説明しない場合があるんだ。そう、いわゆるプログラミングの仕方というやつだ。プログラミングの仕方ってどの言語も基本的なところはほとんど同じなので、他言語学習済みの人向けの入門書ではそういうのが特に多い。プログラミング初心者向けか、その言語初心者向けかは大きく異なるってことは入門書を選ぶときに一番注意すべきことなんだ。良い本は前書きにその旨が書いてあるので読んでおくと良いよ。(逆に、それすら書いてない本は悪い本だから避けた方がいいね。)

さて、プログラミングって実は二段階に分かれている。無意識にしている人もいるけど、ほとんどの人は二段階で思考を行っている。それは次の二つだ。

1. 設計

2. コーディング

プログラミングとは「何かしらの問題を解決するプログラムを書くこと」だ。単純なHello, world.も「Hello, world.と表示する」という問題を解決することに過ぎない。まず「設計」において、問題を解決するための戦略を立てることから始める。問題を分析し、用意された道具でどう解決するかだ。それが設計だ。設計が終われば、次はコーディングだ。設計は作戦のようなもので、コーディングは実際の行動だ。設計したとおりにプログラムを実際に書くことだ。コーディングが終われば、テストや実行があるけど、それはまた次の話だ。

よく、いきなりコーディングをし始める人って見たことあるかと思う。でも、勘違いしないで欲しい。それができる人は設計をしていないのでは無くて、頭の中で、すでに設計しているだけに過ぎないんだ。設計は別に何かに書かなければならないと言うことは無いんだ。ただ、規模が大きくなると、一度に頭の中で整理なんてできないから、何かに書くようにするだけに過ぎないんだ。

うまくプログラミングができないって人のほとんどはこの設計の過程が抜けていて、いきなりコーディングから始めようとしている場合が多いんだ。それじゃ誰だってうまくいくはずが無い。なので、最初はこの設計を意識して欲しい。まずは、問題を解決するための戦略を立てるんだ。どうやって解くのか、そのために何が必要か、何と何を組み合わせれば良いのかを考えるんだ。

設計の仕方はどうするかは…。うん、それを説明するととても長くなるので、今日は割愛するよ。

才能の限界。

関連した質問

解決済 回答:3 / クリップ:1
プログラミング初心者の学習支援について
私は現在，卒業研究にてプログラミング初…

受付中 回答:3 / クリップ:0
if文について
細かい事なのですが、 if(条件) a=b; 単文…

受付中 回答:2 / クリップ:0
参照について
皆様、質問がございます、宜しくお願い申…

解決済 回答:1 / クリップ:0
Swiftの定義ファイルについて
Xcode上で調べたい単語をコマンド+クリッ…

前の文章で「設計」と「コーディング」の話をしたけど、実は「コーディング」はそれほど難しいことじゃないんだ。ただの暗記だからね。つまり、設計の内容を翻訳してコードに落とし込むというただの作業だ。覚えれば誰でもできるし、あまり難しいことじゃないから、コーディングしかできないプログラマーを揶揄して、コーダーと呼ぶときがあるんだ。でも、これは別に悪いことじゃない。設計はそれなりの経験や知識がないと良いものができないから、経験の浅い新人プログラマーにさせたりなんかはしないんだ。仕事なんかだと、誰でも初めはコーディングだけしかさせてもらえないんだ。

ここでただ作業しているだけで終わるか、「この問題を解決するにはこういうふうに設計すれば良いのか」と学んでいくのかで、その人の未来が決まるんだ。つまり、設計をさせるに値するかだ。最初は、1〜10まで設計されたもののをその通りにコーディングしろと言われうけど、そのうち、だんだん歯抜けにされて、最後は1と10しか与えられなくなるんだ。そう、抜けている部分を自分で設計して、コーディングできるかが問われるんだ。

「設計」は実は凄く大きいんだ。小さなプログラムだとそうでも無いけど、大規模なシステムだと、ネットワーク環境やハードウェアを含めた全体から始まって、各サーバの構成とそこで動かすアプリケーション、各アプリケーションの役割、各アプリケーションにおける全体の構成(モジュールとか)、各モジュールの役割やクラス・インターフェース、各クラスにおけるメソッドの内容、そして、具体的なコードの設計と段階的に落とし込む必要があるんだ。最初の方はもうプログラマーでは無くて、SEとかの仕事だから、気にしなくて良い。真ん中のアプリケーションやモジュールの設計あたりぐらいからチーフプログラマーと言う名のSE(？)の仕事だ。実際のコードまで行くと、そこはプログラマーが考えなくちゃいけないことになるんだ。

つまりだ。初めに小さな部品の設計から初めて、それがうまくできるようであれば、だんだん大きなものを任されるという仕組みだ。なので、「設計」ができなくてもそれほど悲観する必要は無いんだ。コーディングが無くなるわけでは無いので、仕事も無くならないんだ。(給料が上がるか、出世できるかは別の話だけどね)

さて、この「設計」部分が結構才能に依存するんだ。勉強すればある程度はできるようにはなるんだけど、数学の問題を解くのに近いものがあって、どんなに勉強しても数学が苦手な人は苦手なままで終わってしまうことがあるのと同じで、できない人はいつまで経ってもできないままなんだ。僕はこれはその人の才能の限界だと思っている。

才能の限界を知ることは悪いことじゃないんだ。僕だって文章を書く才能が無いのに、小説家になりたいなんて思ったことがあったさ(黒歴史)。でも、才能なんてこれっぽっちも無かったから、その道に進むと言う選択肢は取らなかった。もし、限界を知ったら、逆に何ができるかを考えるんだ。設計は苦手だけど、コーディングだけならトップレベルという人だっているし、営業などの別の方面に移っても良いかもしれない。もし、文章を書く方が得意なら、ユーザードキュメント作成とかそういった方面だってあるんだ。

ただね、すぐに限界だと決めることだけは辞めて欲しいと思う。今まではやり方が悪かっただけで、苦手だと思い込んでいるだけかも知れないからね。まずは、チャレンジすることだ。色んなアドバイスがteratailにはあるし、それを一つ一つ試して言ってからでも全然遅くは無いんだ。

とりとめも無い文章になっちゃったけど、では「設計」の話を…。おっと、時間が来てしまったので、また今度にするよ。

2016/05/30 07:35 投稿 raccy
React.js総合1位

コメントを投稿する

ベストアンサー

6

こんにちは。

プログラマ2年目です。
私もプログラミング苦手でした。
あ、今もそんなに得意というわけではないですが、前よりは好きになりました。

そんな私が入社して1年間、以下のことを気を付けてやってみました。

- プログラムを書く前に、処理の流れをコメントで書く
  先輩から真っ先に言われた事ですが、まずはコメントで処理の内容を書くこと。
  //ループ開始
  // 処理1
  // 処理2
  //ループ終了
  こんな感じですかね？実際はもう少し詳細に書いてますが。
  コメントを書かないときと比べると、だいぶ書けるようになりました。

- 仕様を整理する
  個人的な仕様書もどきを作っています。

「ループする」「Nullだったら処理なし」みたいな自分が分かりやすい言葉で。
先輩に聞くときも、どこまで分かっているかが伝えやすかったです。

誰かのプログラムを読んだり、自分でも書いているうちに、少しずつ出来ることが増えていくはずです。
お互い頑張りましょ！

2016/05/30 11:43 投稿 stereo_code
score 47

jawa 2016/05/31 09:11 編集

手を付けやすいように最初に処理の流れをコメントで書いておくのは自分もよくやります。
あと、実装方法がわからない・処理内容が確定していない・単にいまつくるのが面倒なので後回しにしたい、などの場合にも同様にコメントで処理内容だけ記述して置いたりします。
```
//TODO：（実装したい内容）
```

stereo_code 2016/06/03 16:19

jawaさま

私もよくTODOつけてます！
やるべきことが一目で分かるので、良いですよね。

shironegi 2016/06/03 17:44 編集

todo は私も良く使います.
Doxygen のようなドキュメント生成ツールを使ったとき，ルールに沿って付けておくとtodo一覧を生成できたり，個人的には使った事はないですが開発環境のプラグインで一覧にしてくれるようなものもどこかで見たことがあったような気がします.
※最初長いコメント書いてしまったけど場違いなのでけしました.

コメントを投稿する

4

PHP

```php
<?php

/**
 *
 * @param int $age 年齢
```

2016/05/29 18:24 投稿 Kosuke_Shibuya
MySQL総合1位

TetsujiMiwa 2016/05/30 10:56

CとPerlのタグで質問しているのに、PHPで回答しています。
なので、根本的に的はずれな回答と思いマイナス評価を入れました。

コメントを投稿する

3

白いキャンバスに一筆目を入れる時に緊張するのと同じように、何もない状態からプログラムを起こし始めるのは、やはり相応にパワーが必要なことです。
自分は学生時代から含めるとPG経験30年近くなりますが、それでも新規にコードを起こし始めるときには精神的にくるものがあります。
一歩目を踏み出す勇気とでもいうのでしょうか。

そんなとき自分の中に抱えている不安要素は以下のようなものが多いです。

・作りたいもの全体の機能は見えているが、どこからどう手を付けていいのかわからない
⇒全体像（最終的に作りたいもののイメージ）だけわかっているけど具体的になっていない
（詳細設計が詳細まで掘り下げられていない場合など）ということはよくあります。
こういうときは本当に手を付けるのにパワーが必要です。
こんな時はいきなり全体をみず、極々小さい単位の機能に焦点を当ててみるといいかもしれません。

みる、と考えればちょっとクリアなイメージになりませんか？
イメージできるくらい小さな単位に落として考えると作り始めやすいかもしれません。

千里の道も一歩から、踏み出せば案外なんとかなったりするものなのです。

・汎用性を考えると各機能がどういう単位でまとまっていたら使いやすいか、なかなかまとめきれない
⇒最初からある程度まとめられればそれに越したことはありませんが、思い切って後からまとめるという手もあります。
とりあえず関数化は考えずに実装を進めてみて、「あれ？こんなコード前にも書いたなぁ。」と思ったら同じコードを描いていた部分を関数化していくといいかもしれません。
最初から悩んで止まっているしまうよりもずいぶん楽になると思います。
ただ、関数化の単位や実装方法が乱雑になりやすいので、最初からまとめられる部分を把握できているに越したことはないのですが。

・include文やmain関数など、動かし始めるために必要なものがそろっているか不安。
・関数の使い方が正しいか自信がない。
⇒最初は既存コードやwebからコピペして、とにかく動くものを作りましょう。
完全に同じものではなく、部分的に使えるそうなものでも寄せ集めれば使えるものになったりします。
大事なのは、そこで流用してきたコードをただコピペで貼り付けるだけでなく、自分の技術・経験として身に着けていくことです。

自分の経験から勝手にいくつかアドバイスさせていただきました。
我流プログラマなので一般的にお勧めできない部分もあるかもしれませんが、少しでも参考になる部分があれば幸いです。
頑張ってください。

2016/05/30 14:46 投稿 jawa score 473

コメントを投稿する

3

すでにたくさんの有意義な回答があるなかで恐縮ですが、

> 劣等感を感じてなりません。

ここにシンパシーを感じたので少し書かせてください。

私は、大学では情報専攻（就職に役立ちそうと思ったので/Fラン）であったにもかかわらず、コードを見て「こうだよ」と言われればそういうものかとなんとなく上澄みを理解できるものの、自分で１から読むことは満足にできずにいました。ついこの間話題になっていた 新入社員プログラマーですが周りについていけません - 発言広場 のような状態でした（今でもまだその雰囲気はあります）。特に、このリンク先のAnswer No.12へのコメントとして書かれている下記の一節に、大変思い当たるふしがありました。

> こう…九九の表を作りたい、となったとき、「ここでfor文の入れ子構造使えば出来るのでは？」と考えが及ばない感じですかね…。まず「どうすれば出来るの？」となってしまいます。恐らく能力だけでなく想像力も貧困なのだと思います。

これに本当に深く同意する、と話すと、たいていの人は「ああそれはやっぱりプログラミング向いてないね」という反応を示します。でもそういう問題でもない気がするのです。情報専攻じゃなくても、「九九の表を作る」というネストfor文の演習なんて、プログラミングの入門編の授業なんかで、誰でも必ずやらされるものです。だから、その道を志していないような、特にこれまでにプログラミングを経験していないような普通の人が軽々と理解して越えていくスモールステップで既に脱落してしまう自分はいったい何なのだろうと思いました。ほかにも、何度も何度も丁寧にバブルソートの仕組みを説明してもらっても、ちっとも頭に入ってこず、あちこちから「なるほど、こうかー」のような理解への喜びの声がわくのを聞きながら、情けなくて恥ずかしくて悔しくて目頭を熱くするような経験もありました。プログラミングは苦手と公言する人でさえ、説明を聞くとすぐに理解できるようだ、であれば、プログラミングどうこう以前に自分は人間的に欠陥があるのでは、と思い詰めるようにもなりました（実際そうかもしれないが……）。

それが単に自分だけの問題ならよいのですが、不幸にもプログラミングを避けられない職業に就いてしまうと、「だめだ、自分に欠陥がある」などと言ってもいられません。スモールステップでだめなら、タイニーステップでいくしかありませんでした。

アドバイスになるかわかりませんが、以下、自分の勉強方法です。
ここを利用して少しずつ言語に慣れるようにしています：AOJ: Online Programming Challenge
なんとか解けそうな問題に手を付けて、それでわからなかったら先達の解答を見て「なるほど」を増やしていきます。解答の中に、「このクラス何だろう、見たことない」というのがあれば、言語のリファレンスを参照します（私はJavaなので、Javadoc。Cはこれでしょうか？：cppreference.com）。そ

。
並行して教則本・実践本・啓発系（リーダブルコードとか）も、無理のない程度に読み進めます。無理、頭がパンクする、と思ったら読むのは休んで構わないと思います。技術的に必要なときにリファレンスとして実践本を開き、やる気が満ちているときに啓発系を読む、という風にしています。

あとは大事なのは、「こういう仕組みのプログラムが欲しい」となった時にすぐにコードレベルのものを思いつくこと、というよりも、それを実現するためにどんな技術が必要か、調査のためのあたりがつけられる、ということだと思います。すぐにできなくても、自分を責めず、落ち着いて情報収集してください。自分でも「幼稚園児か？」と思うぐらい簡単なところからチャレンジしてください。成功体験が、継続のための力になります。そして、継続は力なり、です。

2016/05/30 15:03 投稿
2016/05/30 16:50 編集

fymartym
score 59

コメントを投稿する

3

私は、プログラムの流れ（実行の流れやデータの変化）が、頭のなかに自分なりのイメージができてからコーディングしています。イメージなので言葉で説明はし難いのですが、あえて言うならピタゴラスイッチの様な仕掛けでしょうか。（あそこまで複雑ではないですが）
それができたら、後は言語に合わせて（文法に則って）コーディングしていきます。
処理が長くて複雑な場合はある程度の機能単位で分割します。どんなに複雑な処理でも分割していけば1つ1つは単純になります。それを組み合わせれば処理は完成します。（組み合わせに苦労するかもしれませんが）

私がプログラムを始めたのは8ビットCPUのアセンブラでしたから、最初はフローチャートを書かないと組むことはできませんでした。でも、それを続けていると、フローチャートを元にした思考に慣れてくるので（脳のシナプシスが強化されるのでしょうね）、細かいフローチャートは書かなくても流れやイメージが自然と浮かんでくるようになります。

よく1万時間の法則って言いますけど、ある程度経験を積まないとできないことは多々あります。プログラミングも例外ではないと思います。短時間でデキる人はそれこそ天才か頭のコロックサイクルが我々より速い人でしょうね。

2016/05/30 17:05 投稿

PineMatsu
score 437

コメントを投稿する

1

こんにちは。

> ソースコードを見れば、何となくですがどうゆう動きがするのかは理解できる

ということであれば、大丈夫です。
手も足も出ないと感じる時は、概ね下記のどちらかの状態に陥っていると思います。

①難しすぎる課題に挑戦している
初心者にとって何が難しすぎるのかは個人差が激しい部分です。
それまでの人生経験の違いですっと入れる人もいれば、中々入れない人もいます。
そのような時は、周りの人に付いて行くことに拘らず、まずは簡単なことにチャレンジすると良いです。
簡単なことで良いので、たくさんチャレンジしていれば、ある日突然「コツ」が掴めます。
そうなれば直ぐに他の人に追いつけますよ。焦る必要はありません。

②もしくは、恐すぎて手がでていない
なんとなくコンピュータを壊しそうな気がして、色々やって見れない状態です。
プログラミングは独特な世界なので、色々やってみてコツを掴まないことには先へ進めません。
色々やってみることに恐れを感じて、クリック一つするのに躊躇する人もいます。

しかし、管理者権限でログインしていなければ事実上壊せませんので安心して良いです。
もし、使っているのが学校の授業で使うコンピュータでしたら、学生は管理者権限でログインできないはずですので安心して良いです。

2016/05/29 17:53 投稿

Chironian
C総合1位
score 4644

1

これればかりは慣れるしかないですね。
私の場合は、サンプルコードやチュートリアルなど、他の人が作ったプログラムを片っ端から入力し、それを改造したり、あるいは真似して作ってみたりして理解を深めていったような気がします。
そうすれば、そのうち、こういうときはこういう風に作れば良いのだな、ということが判ってくると思います。

2016/05/29 18:21 投稿
catsforepaw
score 2576

コメントを投稿する

1

好きこそ物の上手なれ

2016/05/29 23:30 投稿
ttyp03
score 868

コメントを投稿する

1

趣味でプログラミングを始めたころは、よく世の中にあるゲームなどのようなものを
いきなり作ろうとしてしまって、何から手をつけていいかわからないということはよくあることです。
実際話を聞いてみると職業プログラマでも、工数や技術的な問題で難しいものも多々あります。

基本的には、自分で理解した範囲内のものしか作れませんので、書籍やネットなどで学びながら
少しずつできることを増やしていくことになると思います。焦らずこつこつ積み上げていくしかありません。

2016/05/30 09:55 投稿
2016/05/30 09:56 編集
okinaka3
score 68

コメントを投稿する

1

プログラムのフローは

- 順次進行（上から順番に処理）
- 繰り返し
- 条件分岐

の3つ、データの扱いは

- 選択
- 更新
- 追加
- 削除

の4つ。
あらゆる処理は基本この3x4の組み合わせで表現されます。

ということで、

- やりたいことを上記3x4の組み合わせで整理する
- それらを順にその言語で実装する方法を学ぶ

といいんじゃないでしょうか。

コメントを投稿する

1

気軽にドンドン、コードを書いていけばいい、ように思えます。
本当に気楽に一歩目を踏み出してもらえばいい、と思います。
そういう意味では、Perl はすごくいい選択のように思えます。
まずは動かしてみては、いかがでしょうか?
ぼくは、18年前に 初めて Perlのコードを書きましたが、いま読み直してみると、本当にひどいコードです。
いま思うととても懐かしく思えます。

2016/05/30 13:11 投稿  yoichiro_ito
score 45

コメントを投稿する

1

こんにちは。プログラミング歴10年のフリーのSEです。
色々な方からご回答があるようなので、私は**自分の実体験**を交えてアドバイスを書かせて頂きます。

私も学生時代に最初は何から手を付ければ良いのか全く分からず、暫くぼぉ～っとしてて先生に怒られた記憶があります。（笑）
確かに、当時は作ろうとしているプログラムを「自分の頭の中」では理解出来ている！とそう思っていました。
ですが、今思うとその「情報」にはまだまだ不十分なところがあったのです。
まぁ、今まで自分で作った経験が無いのですから、具体的な動作なんて想像できるわけないですよね？

では、どうすれば良いか？

プログラミングをするには、どういう風にコードを組めばどういう処理が発生するのか？という**「想像力」**が必要です。
そして、その「想像力」で作った小さな**「パーツ」**をたくさんくっつけ貼っつけして、ようやく本来の動作ができる**「プログラム」**を形成されます。
例えるなら、こんな具合です。
・「変数」や「命令式」は**【部品】**
・「想像力」はその**【使い道】**
・プログラムはそれらを組み合わせた**【アイディア商品】**

では、その「想像力」を手っ取り早く養うには、どうすれば良いか？
一番手っ取り早いのは、みなさんが仰るとおり「実践経験」なのですが、ここでは違う方法をご紹介します。

～下準備編～

最初に、ソースコードはなんとなく読めるようなので、「変数」や「命令式」の知識はあるものとします。
では、何をするかというと、色々な方のソースコードを読んで下さい。（理解できるなら他言語でも可）
但し、ただ処理を追って解読するだけじゃダメです。

> **ちゃんと処理全体を把握して、どういう目的があってこの処理をしているのか？**
> **そして、どうしてこういう書き方をしているのか？を必ず「想像」しながら解読して下さい。**

また、可能であれば**上記思考回路を癖にして下さい。**
そうすれば、自ずと想像力も身につきます。

目安としては、自分でこのパーツはこうしたらできるよね～という感じで想像できるようになったら次のステップです。

～実践編～

最初は小さなプログラムでも全然構いません。
例えば、**電卓を作るとしたら、どんなパーツが必要ですか？**
私なら最低でも「入力」「計算」「結果表示」のパーツが必要だと考えます。
初心者の場合は、もっと細かくパーツ分けした方が良いですね。

次に、実際に実装する上で重要なことは、次のとおりです。

～重要項目～

1. **過去に自分が書いたコードの経過を保存しましょう！**
   そうすることで、後で見返したときに、自分の思考過程や癖が分かったり、同じ間違いをしなくなったりするものです。
   人によっては無駄と思われるかもしれませんが、自分は成長速度が上がりました。

2. **最初は自分の知識だけでコーディングすること。**
   作業効率は後からついてきます。
   最初は多少時間が掛かっても、コードが美しくなくても自力でやりましょう！

3. それでも分からなければ**ネットや参考書を元に「アレンジ」して使うこと。**
   最初から「コピー」をしていては、その処理に対する理解度が低いままです。
   また、いつまで経っても短時間でのコーディングができません。

以上が、私からのアドバイスとなります。
どこかの記事みたいになってしまって大変恐縮ですが、少しでも読んで頂けた方の糧となれれば幸いです。
長文失礼いたしました。

2016/05/31 17:46 投稿 doraeclipse score 4

コメントを投稿する

0

習うより慣れろ！ でしょうか。
どれくらいやってらっしゃるのか分かりませんが、
地道にやってるとだんだん見えるようになりますよ。
あと、プログラミングに対するモチベーション大事。
どうしても集中できない時は、いったん別のことをして
ノリがよくなったところで取り組むのもいいかと。
そのへんは普通の勉強と同じと考えていいと思います。
まずは簡単なことから始めてみましょう。:-)

2016/05/29 16:44 投稿 takasima20 score 1150

コメントを投稿する

0

PCを使う上で自分がちょっと欲しいなと思う簡単な機能を実現するプログラムから作っていくのが良いと思います。
完成品のビジョンがはっきりしていて、作った後にそれ自体が報酬になり、なおかつ誰も口を入れないため、かなり自分に優しいプログラミングできます。

ただ、根本的な問題としてプログラミングが辛いと思うのでしたら別の方法を考えたほうが幸せになれると思います。
もし、職業としてプログラミングをしようと考えているなら迷わず他の道を選ぶべきです。プログラミングは適性がありますので無理に頑張っても辛いだけです。
もし、何かを実現するためにプログラミングが必要だとしたら、他のアプリケーションやサービスで実現できないか考えるべきです。
プログラミングは手段であって目的ではないですから。

2016/05/29 16:44 投稿 oskbt score 331

コメントを投稿する

プログラミングをする際、フローチャートを作成していますか？
古い考え方かもしれませんが、フローチャートを作成することで以下のようなことがプログラミング前

- どのような処理が必要か
- どのようは手順で処理をするべきか
- どのような処理パターンがあるのか

必要な機能に対して、フローチャートがすぐに思いつくようになるまで、フローチャートの作成をお勧めします。

2016/05/29 16:45 投稿

takyafumin
score 863

コメントを投稿する

0

右も左も解らない間は、ぶっちゃけ、習うより慣れろ、
初心者の題材とされる、「電卓」「ＰＡＩＮＴ」「ＤＲＡＷ」は、
動く物が各ＯＳにあったりすので、
その動きや、機能を見ながら、作成していくことができます。
コードをコピーしてしまっては、何にも成らないので、
コードはコピーせず、機能や、画面を１つ１つ実装していきます。
その過程で不明点は、可能な限り自身で調査、検証を行います。
⇒概念、単語、意味が解らない場合は、「～とは」で検索
⇒グーグル検索は、ログインして使ってあげる、解ってきたらカスタマイズすると、
　徐々に好みの情報が多く集まって来るようになるかもしれません。
　または、「ほんやくさいとまとめ」などの、まとめて検索出来るサイトを
　使ってみたり。
⇒ここで、解らない部分を、調べもしないで、
 すぐに掲示板の世話になってしまうと
 調査、検証能力が身に付きません。（掲示板依存症患者へまっしぐら）
 実装方法や、機能が不明な場合は、コードを貰うのではなく、
 調べ方や、考え方を掲示板で聞くようにするなど、
 掲示板の使い方も、考えて使ってください。
’
以下は戯言。
’
対象を分析し、分解する能力、
分析内容の単純化、普遍化
対象の構成要素を、イメージする能力
⇒これらは、何も無くても、街中の物でできます。
情報収集能力
⇒２次情報、１次情報を集める能力
 ～リファレンスを読み込んだり、
 各種サイトを渉猟して、情報の引出を増やします。
’
仕様書、事象を明示、簡易化した物を作る為に、
箇条書きが出来るようになる事。
日本語で、正しく周囲に、内容、仕様が伝えられる様になる事。
’
作りたいもの、イメージしたものが、頭の中に入っていて
手順を踏んで、項目毎に矛盾無く頭の中で、組み立てできれば、
コード書きは、只の作業です。
’
コード書き、極簡単に書くと
１．機能確認プログラム、デモコード、テストコード
２．実証コード
３．製品コード
４．製品コードリリース後に判明した、バグ修正更新準備
各段階で、デバッグ、仕様変更、修正が入ります。

2016/05/29 18:16 投稿
2016/05/29 18:35 編集

daive
score 1053

コメントを投稿する

ソースコードを書くとなると本当に何をしたらいいのかわからず

「設計」の仕方ではなく、純粋に「どういうコードを書けばいいのかわからない」という質問と理解して私なりの簡単なアドバイスを。。。

どのような処理をさせるかは決まっていても、そこに向かうまでの入力の手続きや表示などの出力の仕方、内部での演算処理。。。 目的や規模によって必要な処理のバリエーションも多彩で、何から考えていいのかも経験や知識の蓄積がないと簡単には決まりません。

では、この「経験や知識の蓄積」はどう穴埋めしていくかというと、自分がやろうとしている事に近いことをしている既存のプログラム(ツールやサービスの機能)をネットなどで調べて、できるだけそれ自体なければ類似する処理をするソースコードを探してどんな事をどのように実現しているのか…ということを数多く読み取るのが近道です。

この作業を進める上で、必要かつ重要なことは、

「処理の流れを分析し、類似する処理を余所から探し出し、その内容を目的と比較しながら吟味・分析し、必要に応じてアレンジする。最終的に得られたソースコードだけでなく過程で得られた情報も含めて概要を知識やメモとしてストックし、次のための糧にする。」というループです。

これは、プログラミングに限ったことではありません。

流れとしては以下の様になります。

1. やろうとしていることを、順番にできるだけ細かくパズルの「部品」のように分解する。

2. 分解した部品ごとに、似たような処理をしているソースコードをネットや書籍、過去の自分のソースコードから探す。

3. コピペしてから自分の目的に合わせてアレンジしてコードを書く。

4. うまく動かない時はネットや書籍で調べて修正する。その過程で調べた知識もストックしておく。

基本的には、ジグソーパズルのピースを探して組み上げる作業や、レゴブロックを組み上げることと同じです。ピースやブロックの種類が目的に応じてべらぼうに多くて探したり覚えたりが大変なだけですww

この流れの繰り返しの蓄積が「慣れる」ということです。

簡単ですが、アドバイスとなれば幸いです。

2016/05/31 14:01 投稿 AnMoreNight score 17

コメントを投稿する

0

[プログラミング] を [作曲] とか [文書執筆] とか ［スポーツ] とかに置き換えて考えてみるとよいとおもいます。

どんなことでもはじめは皆 初心者です。
どんな風に上達していくのか？
それが好きであること、練習を重ねること、他の人のものを見ること, ...など。

2016/06/02 06:48 投稿 katoy ユーザーランキング総合2位

コメントを投稿する

あなたの回答

tips

回答を入力してください※ 回答を間違えた場合は編集機能をご利用ください。※ 回答するのに必要な追加説明は追記依頼を、他の回答への返答はコメントを利用してください。

and or 質問する 1 HidetosiSirai 1

回答を投稿する

関連した質問

解決済 **言語の性格とプログラミング「感覚」の相関について**
最近、Scalaの練習をしています (Apache Sparkを研究しようと思って)。 Scala、コンパイルが遅いですね。それで…

解決済 **クラス,関数,変数のネーミングについて**
私は最近自分で書いたプログラムを他人に見せることが多くなってきました。 そこで、今までは適当につけがちだった…

解決済 **PHPで演算子の組み込み場所について**
PHPで + や / などといった演算子は、どのように言語構造に組み込まれているのか 実際に確認をしたいのですが、ど…

受付中 **コーディングについて**
仕事でプログラマ兼雑用をしております。 事務職志望だったのがなぜかシステム開発部に異動となり、 知識が全く無い…

同じタグがついた質問を見る

C Perl

PAGE TOP