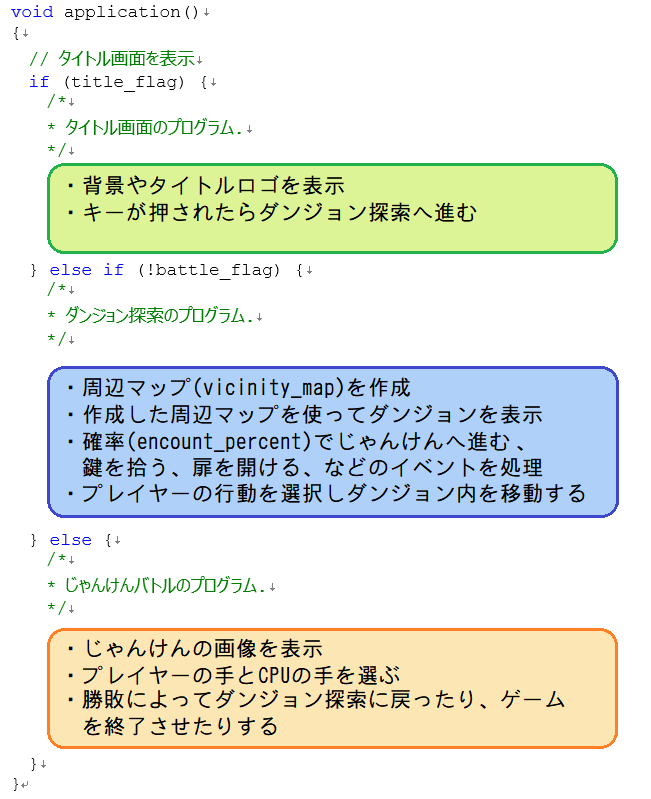
[OpenGL 2D 2018 第06回]

プログラムは上から順番に実行される

「順番に実行されるとどうなる？」  
「知らんのか。」  
「プログラムが動く。」

# 全体を見渡す

これまでみなさんに作成してもらったじゃんけんゲームのプログラムは、大きく3つの部分に分かれています。これを図にすると次のようになるでしょう。



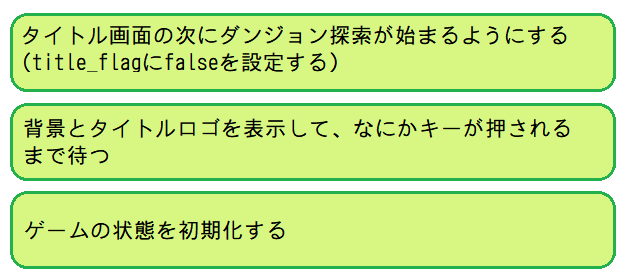
title\_flagとbattle\_flagという2つの変数によって、タイトル画面、ダンジョン探索、じゃんけんバトルのうちいずれかが実行されます。2つの変数の値を変更することで、プログラムのどの部分を実行するかを制御することができます。

このあと、この3つの部分のそれぞれについて、プログラムの内容を改めて追いかけていきます。

# タイトル画面

## タイトル画面プログラムの構造

タイトル画面のプログラムは3つの部分に分けられます。タイトルフラグにfalseを設定して、ダンジョン探索が実行されるようにする部分と、背景やロゴを表示してキー入力を待つ部分、そして画面を消したりゲームの状態を初期化して、ダンジョン探索に進む準備をする部分です。



タイトル画面のプログラムは基本どおり上から順番に実行されます。ですから、いちばん理解しやすいのではないかと思います。

タイトル画面のプログラムでは、最初にtitle\_flagにfalseを設定しています。これによって、次にダンジョン探索プログラムが実行されるようにしています。

**[補足]**  
title\_flagにfalseを設定する部分は、「ゲームの状態を初期化する」プログラムで一緒に行うべきです。しかし、これが初めて登場したのは第03回です。その頃、みなさんは変数についてまだよく理解していなかったと思います。ですから、うっかりこの変数をいじってしまってじゃんけんが始まらない、ということがないように、前に書いておいたのです。

**[課題1]**  
wait\_any\_key関数でキー入力を待つのではなく、select関数を使って「はじめる」と「おわる」の2つの選択肢を表示してみましょう。「はじめる」が選ばれたらダンジョン探索へ進み、「おわる」が選ばれたらquit関数を使ってゲームを終了させてください。

# ダンジョン探索

## ダンジョン探索プログラムの構造

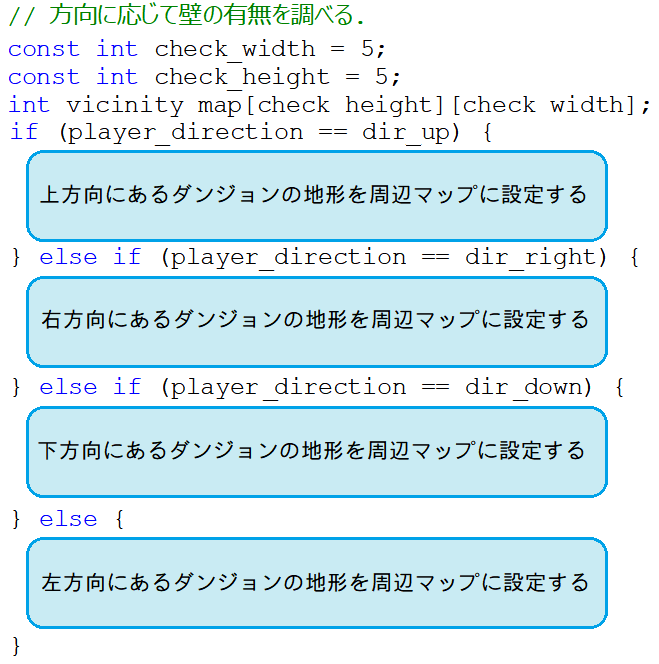
ダンジョン探索のプログラムは、今回作成してもらったプログラムの中で、もっとも複雑な部分です。このプログラムは、さらに4つの部分に分けられます。

1. 周辺マップ(vicinity\_map(ゔぃしにてぃ・まっぷ))を作成する。
2. 周辺マップを参考にして、プレイヤーから見たダンジョンの画像を作成する。
3. 確率(encount\_percent(えんかうんと・ぱーせんと))でじゃんけんを発生させたり、鍵を拾う、鍵のかかった扉を開ける、といったイベントを処理する。
4. プレイヤーに行動を選択してもらい、それにしたがってプレイヤーの向きや位置を変更する。

また、各プログラムは何個ものfor文やif文を組み合わせて作られています。

## 周辺マップを作成するプログラムの構造

周辺マップ(vicinity\_map(ゔぃしにてぃ・まっぷ))は「プレイヤーから見える範囲にある壁や扉の情報」です。「周辺マップを作成するプログラム」は、プレイヤーが向いている方向にある壁や扉などの情報を、周辺マップに設定します。

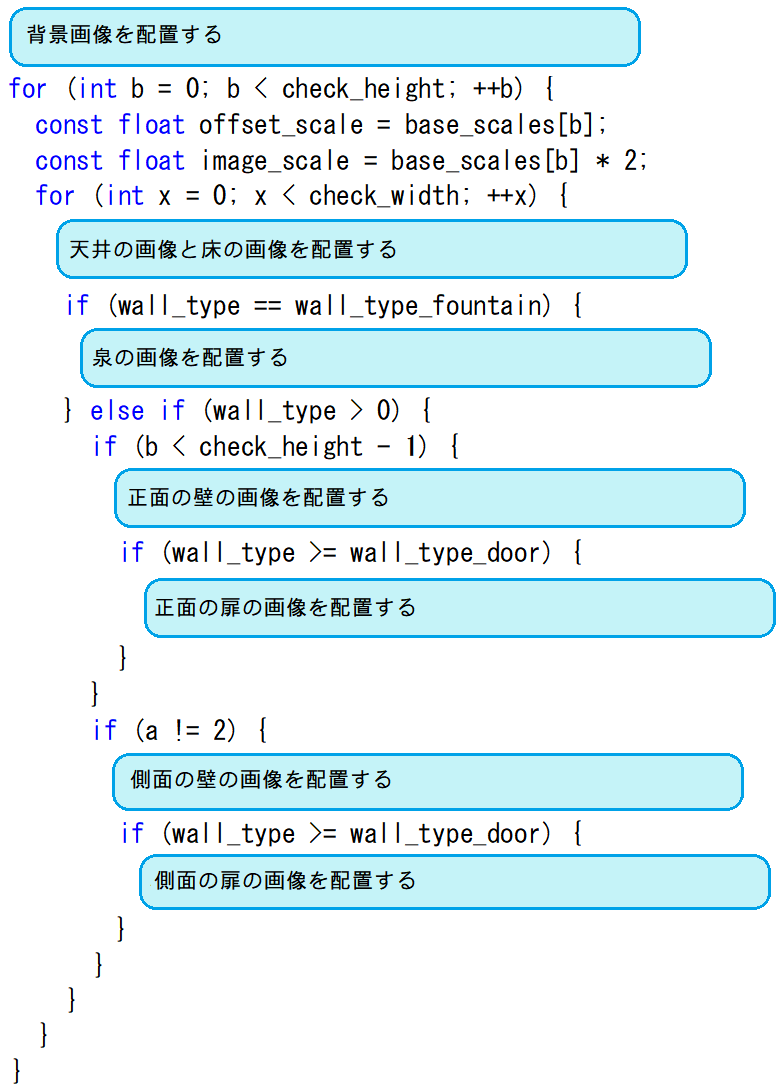


プレイヤーの向きによって見える範囲が違いますので(みなさんだって前を向いているときと後ろを向いているときで見えるものが違いますよね)、向きごとに異なるプログラムが実行されるようになっています。そして、向きによって違うプログラムを実行させるためにif文を使っています。また、周辺マップは2次元配列になっていますから、繰り返し処理のために2つのfor文を使っています。

**[課題2]**  
for文の繰り返し判定では「5」とループ回数を直接書いています。これをcheck\_widthとcheck\_heightを使うように書き換えてください。ダンジョンの画像を作成するプログラムのfor文を参考にしてください。

## ダンジョンの画像を作成するプログラムの構造

「ダンジョンの画像を作成するプログラム」は、「周辺マップを作成するプログラム」が作ってくれた周辺マップを参考にしてダンジョンの画像を表示します。



周辺マップは2次元配列なので、ここでも繰り返し処理のために2つのfor文を使っています。2つのfor文のさらに内側には、天井や床や壁、それに扉や泉といった画像を表示するプログラムがあります。どの画像を表示するかは周辺マップに設定された情報によって決まります。そして、情報に対応する画像を表示するためにif文を使っています。また、画像をどこに表示するかは配列によって決めています。

**[補足]**  
これらの配列の内容は、実際には計算によって導かれています。例えば正面の壁の位置はwall\_front\_offsets(うぉーる・ふろんと・おふせっつ)という配列に設定されていますね。この値は正面向きの壁の画像をいちばん手前に表示したときの大きさを基準にしています。正面向きの壁の画像の大きさは512x512です。画像の拡大率を決めるimage\_scale(いめーじ・すけーる)はbase\_scales(べーす・すけーるず)の2倍で、base\_scalesのいちばん手前の位置に対応する拡大率は1です。つまり、正面向きの壁をいちばん手前に表示したときの大きさは、

縦: 512 x 2 x 1 = 1024  
横: 512 x 2 x 1 = 1024

ということになります。そして、画像を隙間なく並べるには画像の大きさだけ横にずらせばいいですよね。画面の中心の座標は(0, 0)ですから、左に1マスずらすなら(-1024, 0)、右に1マスずらすなら(1024, 0)となります。  
また、周辺マップに記録する範囲はプレイヤーの左右2マス分ですから、表示するマス目の位置は-2～+2までです。この2という数値は「check\_width / 2」とすれば得られます。さらに、プログラムでは横のマス目の位置は変数aで表されていますから、

画像のX座標 = 1024 x (-check\_width / 2 + a) x image\_scale

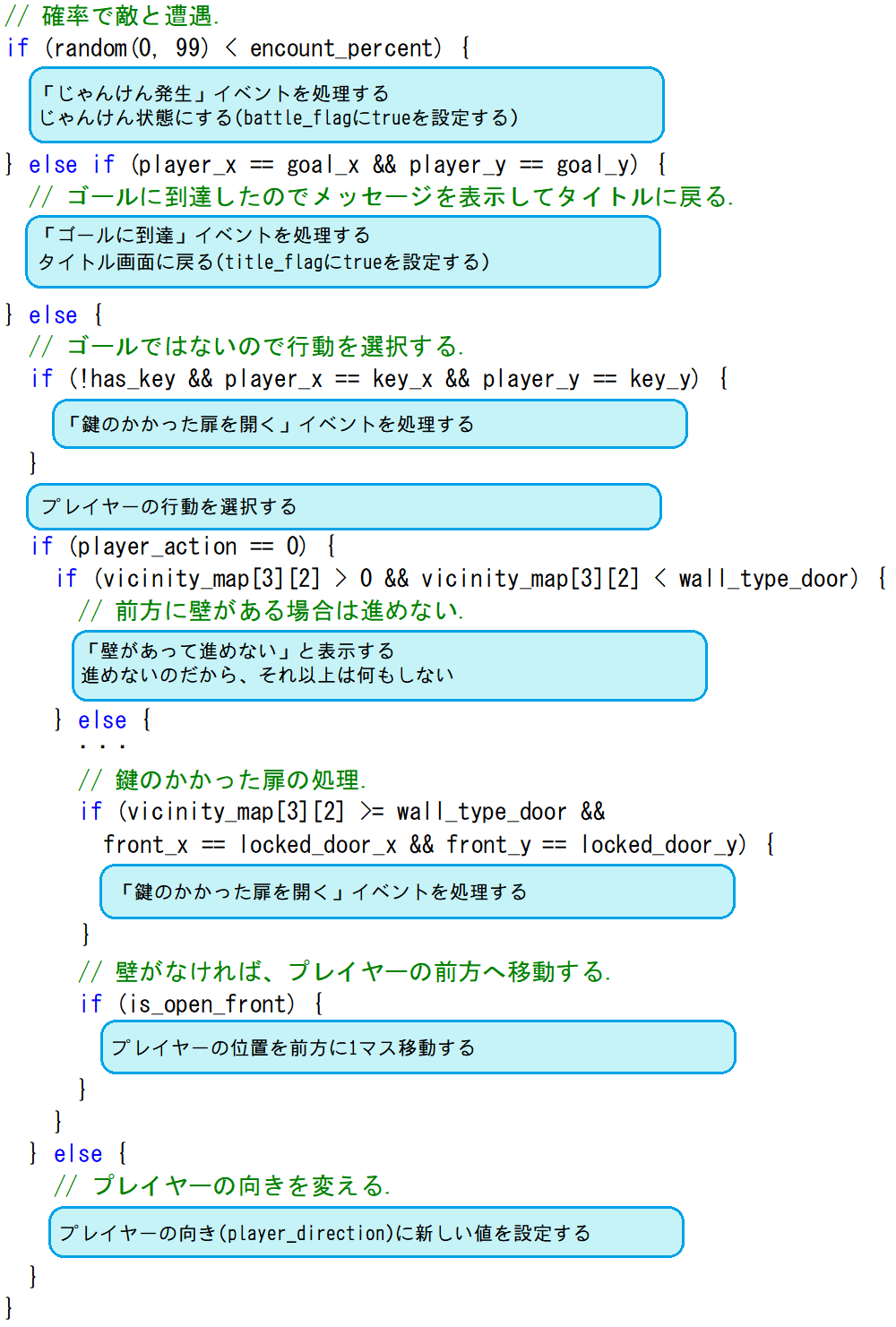
という式が成り立ちます。これを事前に計算し、配列にしたものがwall\_front\_offsetsというわけです。

**[課題3]**  
wall\_front\_offsetsを計算式で置き換えてください。同様にfloor\_object\_offsets, wall\_side\_offsets, ceil\_offsets, floor\_offsetsなども、計算式で置き換えられるか試してみてください。

## ダンジョン内で起きる出来事(イベント)を処理するプログラム

ダンジョンの画像を表示したら、次はダンジョン内で起きる出来事(以後「イベント」と呼びます)を処理するプログラムが実行されます。イベントの性質によって、プレイヤーの行動選択より前に起きたり、あとに起きたりします。イベントが発生するかどうかを判定するためにif文を使っています。

どのイベントをいつ起こすかは、ゲームの内容を考える人が決めます。  
「じゃんけんの発生」、「鍵を拾う」というイベントは、画面を表示してからすぐに起きてほしかったので、行動選択より前で起きるようにしました。また、「鍵のかかった扉を開ける」というイベントは、行動選択で「前進」を選んだときだけ起こしたかったので、行動選択よりあとで起きるようにしました。



プログラムは順番に実行されるものなので、何がいつ実行されるかということはとても重要です。順番を間違えると思ったとおりの動作にならなくなってしまいます。例えば、「鍵のかかった扉を開ける」というイベントを行動選択より前に実行したとしましょう。そうすると、行動選択の「前進」をさまたげるイベントがないので、鍵をかけたにもかかわらず「前進」することができてしまいます。

## 行動選択を行いプレイヤーの状態を変更するプログラム

プレイヤーの行動選択はselect関数で行っています。  
ダンジョン探索プログラムでは、4つの行動を「前進」と「方向転換」の2種類に分けてプログラムしています。どちらの行動なのかを判定するためにif文を使っています。

イベントを処理するプログラムで説明したように、「前進」が選ばれた場合は「鍵のかかった扉を開ける」というイベントが実行されることがあります。

**[課題4]** 方向転換をif文のみで書いてみてください。

# じゃんけん

## じゃんけんプログラムの構成

このプログラムは次のように4つの部分から構成されます。

1. プレイヤーとCPUの状態や画像を表示する。
2. プレイヤーとCPUの手を選び、手に応じた画像を表示する。
3. 勝負の結果に対応する画像を表示する。
4. 勝負をつづけるかどうかを選択する。

じゃんけんプログラムはみなさんに最初に動かしてもらったプログラムです。そのため、どうしても必要なif文をのぞいては、基本どおりに上から順番に実行されます。

## プレイヤーとCPUの状態や画像を表示するプログラム

じゃんけんを行う準備として、プレイヤーとCPUそれぞれの勝った回数を文字で表示し、「じゃんけん」または「あいこで」画像を表示するプログラムです。  
課題を進めているなら、プレイヤーの体力を表示するプログラムも含まれていると思います。

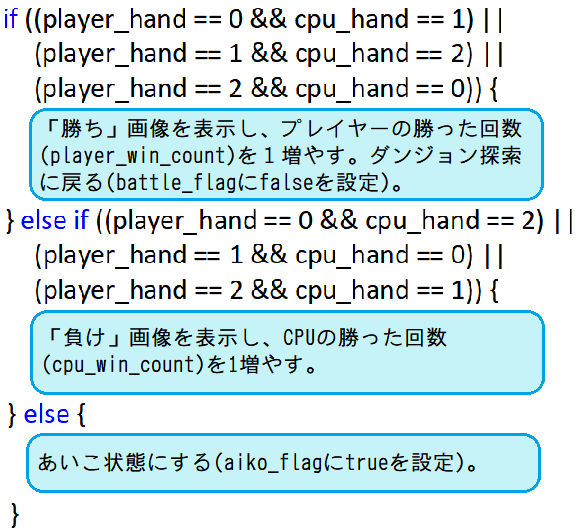
## プレイヤーとCPUの手を選ぶプログラム

プレイヤーの手の選択はselect関数で行っています。CPUの手の選択はrandom関数で行っています。選択肢は「グー」「チョキ」「パー」の3つです(もちろん、改造して4つ以上の手を選べるようになっている人もいるでしょう)。

双方が手を選んだら、選んだ手に対応する画像を表示します。

## 勝負の結果を処理するプログラム

プレイヤーの勝ちなら、プレイヤーの勝った回数を1増やして「勝ち」画像を表示し、ダンジョン探索に戻ります(battle\_flagにfalseを設定)。負けならCPUの勝った回数を1増やして「負け」画像を表示します。どちらでもなければあいこです。また、勝ちと負けの場合はあいこ状態にはなりません(aiko\_flagにfalseを設定する)。あいこの場合はあいこ状態にします(aiko\_flagにtrueを設定する)。



## 勝負を続けるかどうかを選択するプログラム

あいこ状態ではなかったときはプレイヤーに勝負を続けるかどうかを選択してもらいます。ここでもselect関数とif文を使っています。select関数によって「続ける」「やめる」という選択肢を表示し、if文によってどちらの選択肢が選ばれたのかを判定しています。

# まとめ

## プログラムの基本は「上から順番に実行」

「じゃんけん」と「ダンジョン探索」の2つの部分からなるゲームのプログラムを作ってもらいました。  
第01回でお話したように、プログラムは書かれたとおりに、順番に実行されるものです。それはこのじゃんけんゲームも例外ではありません。今後、みなさんはさまざまなプログラムを書いたり読んだりすることになると思います。そのとき、プログラムがどのように実行されるのかで悩んだら、基本に立ち返り「上から順番に実行される」ということを思い出してください。

## ゲームをより面白くする

じゃんけんゲームはひとまず完成しましたが、もっといろいろな要素を入れてより面白くしたい、というひともいることと思います。そこで、いくつか指針を示しておくことにします。

**[じゃんけんの改良]**

* 「相手にも体力パラメータを作る」  
  「体力」を導入する課題がありましたが、同じことをじゃんけんの相手にも行います。  
  先に相手の体力を0にしたほうが勝ちというルールです。
* 「プレイヤーのパラメータを増やす」  
  体力だけでなく、攻撃力や防御力、魔力などを追加します。  
  攻撃力のぶんだけ相手に与えるダメージが増える、防御力のぶんだけ自分の受けるダメージを減る、魔力があれば必殺の魔法が使える、といったルールが思いつきます。
* 「じゃんけんをRPGの戦闘のように変える」  
  「グー、チョキ、パー」を「戦う、魔法、道具」などに置き換えます。  
  そして、手によって勝敗を決めるのではなく、「戦う」を選んだら相手の体力を減らす、といったプログラムに書き換えていきます。

**[ダンジョン探索の改良]**

* 「ダンジョンマップを広くする」  
  現在は8x8の大きさですが、これを16x16の大きさに拡大します。  
  より入り組んだ迷宮を作ることができるでしょう。
* 「イベントを増やす」  
  マップを広くすることで、さまざまなイベントを仕掛けることもできるようになるはずです。  
  マップの別の場所にワープしてしまうマスや、一方通行の扉、斧を投げ入れると女神が出てくる泉など、思いつくままにイベントを追加してみましょう。
* 「壁、天井、床の画像のバリエーションを増やす」  
  そのへんの地面や塀の写真を壁の画像に加工すると、また違ったイメージのダンジョンになります。踏むとダメージを受ける床、深い川になっていて進めない床などを追加してもいいでしょう。また、壁を窓のついた建物風にして、天井を配置しないようにすると、街の中にいるような画面にすることもできます。