プログラムメモ

田中健人

READMEに記述してある通り、公式から配布されたファイルは以下のディレクトリから成る。

* documents:ゲームのルールやデータファイルの形式についての文書が入っています。(プログラムじゃない)
* editor: ゲームに用いるコースを編集するためのソフトウェアが入っています。(プログラムとしては一切関係ない)
* official:レースの実行管理システムが入っています。(バリバリのプログラム)
* player: ゲームを競うプレイヤソフトウェアのサンプルが入っています。(バリバリのプログラム)
* samples: レースコースやレースのログのサンプルが入っています。(sampleのコース、ゲームログでありプログラムではない。別に中を見る必要もない。)
* viewer: レースのログをビジュアライズするビューワのソフトウェアが入っています。(プログラムとしては一切関係ない)

documents内のファイルで関係あるのは、datafiles.pdfとrule-jp.pdfくらいかな。

rule-jp.pdfは郷がslackにあげてくれたから、実質、datafiles.pdfだけ。ゲーム内で扱うデータ構造について説明がある。

editor内のhelp.htmlにこのコース編集ソフトウェアの説明がある。

editor.htmlから、ソフトウェアを動かす。

どんなコースでプレイするか直感的に作成して、ゲーム上で使用できるステージデータを作れる。

samplesディレクトリには、拡張子(.smrjky)のファイル、拡張子(.json)のファイル、拡張子(.racelog)のファイルの3種類のファイルがある。

これらのファイルは先ほど説明したdocuments/datafiles.pdfに書いてあるが、

jsonファイルは編集用コースのサンプルで、

先ほど説明したコース編集ソフトウェアで開くことができ、編集できる。

(もちろんコース編集ソフトウェアで1から作ることもできる。)

smrjkyファイルはコース編集ソフトウェアで完成したコースをゲームが読み込める形に書き込んだもの。

実際にレースの実行管理システムはこのファイルを読み込んでるはず。

racelogファイルはゲームが終わった後出力されるファイルであり、次に説明するviewerディレクトリにあるビューワのソフトウェアで、ゲーム結果を可視化してみれる。

ちなみに、標準であるsample.racelogはsample-course.jsonのコースを書き込んだsample-course.smrjkyでゲームを行ったログ。

viewerディレクトリは、ゲームの結果を可視化するソフトウェアがある。

help.htmlに使い方、viewer.htmlで実際のソフトウェアを起動する。

このソフトウェアからracelogファイルを読み込むことで、そのゲームを見れる。

残りの、officialディレクトリとplayerディレクトリはバリバリのプログラムファイル。

officialディレクトリのプログラムは、C++で書かれたレースの実行管理システムのプログラム。一切書き換える必要はないが、レースの実行管理システムを見て何か気になる処理を確認することがあるかもしれないので、読むことはあるかも。

.vcxprojファイルはこれらのソースをビルドする時の情報を記述したxmlファイル。

今は全く気にしなくていいけど、これらのソースはboostライブラリを用いていて、boostをインストールしてビルドしてから実行しなくちゃいけない、詳しくはREADMEに書いてある。

READMEに書いてある通りにやって動けば今はそれでよし！

と思ったけど、playerディレクトリにあるソースは正にゲームAIのソースで、いわばこれがゲームAIのサンプルなんだけど、これ自分で作った時C++で作ったら、多分自分でこのあたり書かないとだめだね。

最後に、playerディレクトリの説明なんだけど、この中のプログラムって

たぶん

①貪欲法で探索するアルゴリズムのPlayer(c++で書かれてる)

②疑似的なアルファベータ法で探索するアルゴリズムのPlayer(c++で書かれてる)

③ランダム探索するアルゴリズムのPlayer(javaで書かれてる)

④先行探索するアルゴリズムのPlayer(javaで書かれてる)

⑤ランダム探索するアルゴリズムのPlayer(Pythonで書かれてる)

⑥(どんな探索しているか分からん)アルゴリズムのPlayer(Pythonで書かれてる)

の6種類のPlayerのソースが入り混じってない？

それぞれで、実行方法も異なるし、これは今後のタスクで手分けしてそのPlayerの探索手法の調査もふくめてやった方がよさそう。

一番上位のMakefileって走らせても、貪欲法のPlayerの実行しかされない、たぶん

他のPlayerを実行するならMakefile書き換えないといけないのかな

それか直接コンパイルとかをするか

とりあえず、READMEに書かれてる方法のうち、msys2をインストールする方法で環境をととのえて、(gccとか、makeとかいれて、Latexとか、javaとかいれるのメンドイカラMakefileから消して)一番上位のMakefileを無理やり走らせた。(ちゃんと用意してから動かしたほうがいいけど、、)

./official/official samples/sample-course.smrjky player/greedy 0 player/greedy 1 --stdinLogFile0 stdin0.txt --stdinLogFile1 stdin1.txt --stderrLogFile0 stderr0.txt --stderrLogFile1 stderr1.txt > gamelog.json

を実行すると、gamelog.jsonファイルが生成されたので、ビュアーで動かしてみる。

動いた動いた。

これで、動作確認完了。

今回は貪欲アルゴリズムVS貪欲アルゴリズムで動かしたけど、他の組み合わせや、ステージを変えたりすると全然違う結果になりそうで面白い。

これ、いろんなステージでいろんなアルゴリズムのジョッキー動かして、ステージごとの得意不得意とかからメリット・デメリット実感できそう。

細かい探索処理部分は、その後のタスクとして、おおまかなPlayerの共通処理を読んでいって、最弱のジョッキーを作ろう。

~~こっちはJAVAで作る。~~

~~Gamestate.javaは入力データを使いやすい形にまとめてくれてるクラスなのでそのまま使う。~~

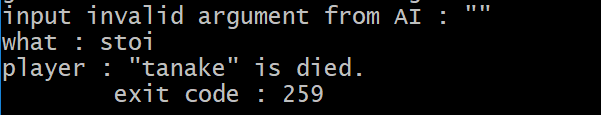
ちょっと待ってね。

とりあえず、壁の判定とかそんなの諸々気にせず

ただただ、真っすぐどんどん加速していくだけのplayerを実装しようとしたんだけど、javaで実装してもc++で実装してもゲーム側に不正判定されて失格にされるんよね。(Pythonは試してない。)(コンパイルはエラー出ずに通る。)

ただ、その理由が分からんくてずっと止まってた。

下にそのエラーを張ります。



JAVAでもc++でも同じエラー。

AIは初期情報を受け取ったら初めに0を送信しなくてはいけない、っていうルールに従って標準出力に0を送ってるんやけど、0として渡されていないんよね。

sampleのコードは出力情報をすべて標準出力として出力してるんやけど、そのあたりについて今未解決で最優先で解決させなくちゃいけない点を記してこのドキュメントは終わらします。

実行時に、標準出力はgame.logにリダイレクトしてる訳だけど、そもそもゲーム側のプログラムにエラー扱いされて、game.logは出力されていない。

標準出力の情報はgame側に渡されないかんのやけど、そのために標準出力の出力先をターミナルじゃなくて、切り替えている可能性があるんよね。

その切り替えを行ってる(指定している部分)はどこだろか？

c++の場合、依存関係の記述にboostを利用していて、それらの情報をxml形式で.vcxprojファイルに記述しているみたいなんやけど、あんまりboost理解してないんだよな、俺ごめんなさい。

ただただ面倒なイメージがあって、、

どのみち、自分らでもboostつかって、c++書かないかんから、boost周りの勉強必須。

けど、Javaの場合は別にboost使ってない(そもそもboostはc系統だけだと思ってる)から、Javaでも同じエラーが出る以上今回のエラーにはあんま関係ないかも。

あるいは、自分で実装したソースはmakefile使わずに、コマンドからコンパイラ指定してコンパイルしたんやけど、そのことが問題なのかも。

どのみち、makefileも自分で作らないかんから、makefileの勉強も必須。

JAVAにしても、c++にしても、sampleのプログラムは、入力情報を変数に割り当てて、利用しているし、これらの操作は共通のクラスでまとめてあって、そのクラスを利用してる。

だけど、情報を受け取ったことを、ゲーム側に通知している処理は別になかったし(正確には俺には見当たらなかった、だけど)、初めに送る必要がある0を送った後は、whileで繰り返してるし、その間ゲーム側と同期してやり取りしている部分の処理も別になかったんだよね、だから、最もシンプルに

ただただ、真っすぐどんどん加速していくだけのplayerを実装する場合、

下のプログラムで十分だと予想したんだけど、



あのエラーが出るんだよね。

エラーの内容的には標準出力の出力先がおかしいからだと思うんだけど、

もしかしたら、同期する処理(ソケット通信みたいに、入力されるまで待つ処理)とかが、俺が見落としてるだけで、あるかも。

てことで、アルゴリズムとか実装する前に、まずはこのあたり解決しないと、そもそも動かせないから、ちょっとみんなで早急に対応したいです。

てことで、俺がやったのは、

運営から配られたファイル群についてのまとめと、

sampleのplayerをコンパイルして動かす、あるいはステージ生成ツールを利用してみる。

ステージの指定方法やゲームの実行方法のまとめ。

だけど、自分で実装したソースはコンパイルはできても、ゲーム側と想定したやり取りを出来てない。

ので、その原因として考えられる可能性を列挙した。