子供からプロまでが楽しめるプログラミングゲームの提案

高見　名越　王　田子

2023 年 1 月 30 日

1　目的

2020 年度，政府は小学生から中学生までの義務教育で，プログラミング教育を必修とした．この政策には，「身近な生活でコンピュータが活用されていることに気付かせること」や，「基礎的なプログラミング的思考と論理的思考力を身に付けること」が目的とされている．

しかし，我々は「アルゴリズムを淡々と理解させるだけでは学生たちのモチベーションに繋がらないのではないか」と考えた．この問題を解決するには，すぐ目に見える実績と達成感に加え，友人と知識をぶつけ合い「○○君より上へ」という競争心が何よりモチベーションになるのではないかと考えた．

本作品は，プログラミングを学ぶ初心者からプログラミングのプロまでの幅広いユーザーに向けて，ゲームの形で楽しくプログラミングを学習および練習できる対戦型パズルゲームを作成する．

2　作品構想

本作品は，ユーザがプログラミングの作成により対戦することを実現する．具体的には，適当なプログラミングを入力することで相手の陣地に攻撃することや自分の陣地を防御することができる．そして、相手の陣地をすべて制覇することを勝利する条件とみなす．

3　独創的な点

現時点のプログラミングツールには，大まかに本格的にプログラミングを行うややこしいツールと，描画機能が含まれるより簡単なツールがある．しかし，本格的なツールにはプログラミング言語を別のところで習得する必要があり，またコンパイラの入出力が文字列や数字だけで面白さも不足であると考えている．また，描画機能を付けるプログラミングでは，面白さは十分であるが描画するために新しい文法を学習する必要がある．その文法はソフトウェア開発などの応用はできない欠点が存在する．

本作品では，ユーザを楽しくプログラミングを学習しながら，現場で使えるプログラミング知識を学ぶことを目指している．

4 詳細

本ゲームアプリは二人のプレイヤーが二台のパソコンを並べ，ローカルネットワーク環境下でプレイすることを想定している．

このゲームは各パソコンに表示されたお互いのマス（メモリー）を, 全て制圧した方が勝ちというルールである. また, 図 1 のようなお互いの陣地は 5x5 のマス（メモリー）で表現され, マス（メモリー）を塗ることで制圧となる．基本的に, 相手のマスを塗るもしくは自分のマスを守るためには，用意されたコンソール上でプログラムコードを入力する必要がある．また，これらのコードは自分陣地内のマス（メモリー）にも書き込むことができ，メモリ内に書き込んだコードは相手に塗りつぶされない限り，一秒間に一回実行され続ける．よって各プレイヤーは，コンソールに書き込む, または自分のメモリに永続的に実行されるコードを設置する，この二つの方法で戦うことになる．入力するプログラムコードにはプログラミング言語と同じく，使用できる自作の関数は下記を予定している．また，それぞれの関数には引数としてメモリの座標を入力することができるように設定する．

ユーザが使用するプログラミング言語は java を基にし，こちらで用意したものである。java でできることをすべてを網羅するのではなく，こちらで用意する関数や繰り返しなどの機能をユーザは使用できる．また，ユーザが記述したコードをパージングする．

- attack(): 相手のメモリーを塗りつぶすために使用し，このゲームの軸となる関数
- connect(): メモリに書き込む際に書き込むメモリを指定するのに使用する関数
- recovery(): 自分陣地内において相手に塗りつぶされたメモリーを無色に戻す関数
- lock(): 自分または相手陣地のメモリーにパスワードを掛ける関数 (パスワードが解除されない限り，色の変更やコードの書き換えはできない)

現時点では，ユーザのプログラミングスキルは変数，計算，演算子，条件分岐，繰り返し，配列と関数の範囲で練習できると想像している．また，実装する時間によって，コメントや再帰関数などの追加も予定している．

また，ユーザの入力したものを文字列として受け取り，その文字列を文法チェックを行う．これによってユーザが入力している内容は正しいかどうかを判断する．相手の陣地をすべて攻略すると勝利とする．また，毎回の対戦でプレイヤーの戦略やゲームの難易度によって書くべきプログラムの量が変わっているが，簡単なバージョン (3×3 の陣地) でシンプルな戦略 (メモリーを一つ一つ攻撃する) を例にすると，プレイヤーが書く必要があることはおおむねに以下で示す.

```
i = 0 から 2 まで
  j = 0 から 2 まで
    memory[i][j] とつながる
    もしそこにパスワードが掛けているなら:
        パスワードの中身を侵入し試す
        ロックを外す
    そのメモリーを攻撃する
場合によって自分のメモリーを保護する（パスワードを掛ける）
```

それ以外に，例えばプレイヤーが for 文などの繰り返しがわかるなら，一つずつ侵入するよりも同時にできることができる．これらの操作によってコーディングの量が減らすことができ，またスキルが上達になるとより効率よくゲームを終わらせることもできる．現在は以上の機能を実装予定しているが，戦略に幅を持たせるため，今後も随時追加予定である．

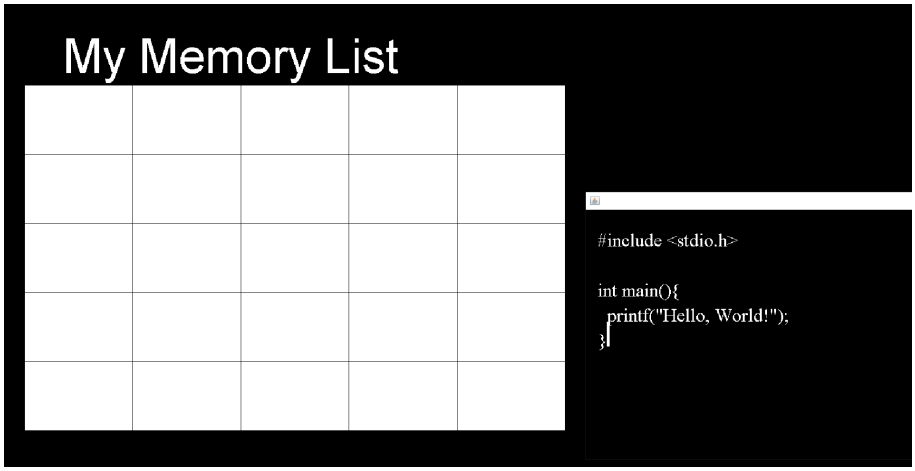

図 1　陣地の画面

5　制作計画

5.1　役割分担

本作品の役割分担以下のように分かれる．

- アプリの基盤作成: 名越
- メニューの作成: 名越
- 自作の関数の完成: 王
- 入力内容をゲーム内に反映: 王
- ゲームの状態を判断: 王
- サーバー側の作成: 田子
- クライアント側の統合: 高見

また，各階段において，デバッグなどは交代で行う．

5.2　スケジュール

本作品について大まかなスケジュールは以下の通りである．

- 九月下旬〜十月上旬: 主旨を決め，企画書を作成
- 十月中旬〜十月下旬: 基盤及びメニューを作成，ゲームの状態判断を作成，クライアント側の作成
- 十一月: 中間発表に向けて準備，入力内容をゲームに反映，クライアント側の作成

- 十二月: 全体的なデバッグ，サーバー側の作成
- 一月: サーバー側の作成，アプリの使用について実験，期末発表に向けて準備

また，毎月末に全員が集まって，進捗と合わせつつアプリ全体の制作方向を調整しながら，細かくデバッグや補足する時間が予定されている．

6　備考

ゲームの構想は全て別紙に纏めている．
現時点で考えられている絵コンテを下記に纏める．

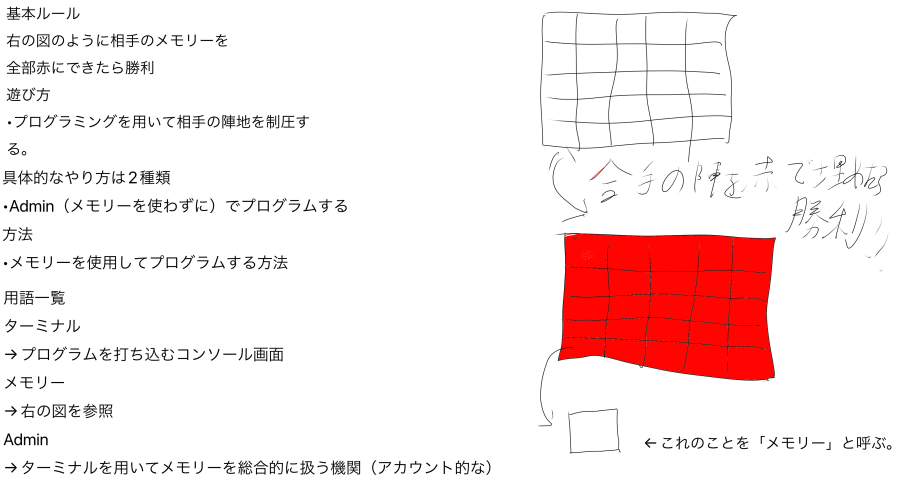

図２　基本ルール

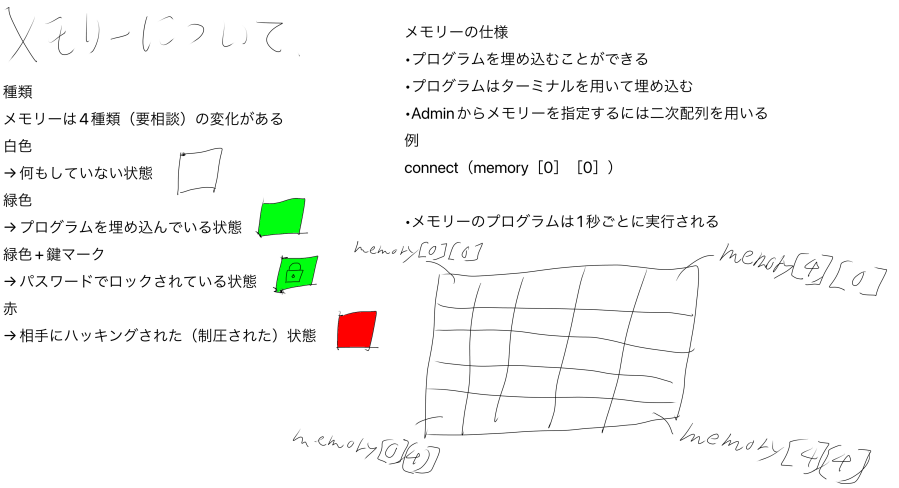

図３　メモリーの仕様

ハッキングについて

制圧の仕方

・attack（）メソッドをメモリー、もしくはAdminのターミナルで使うことで相手のメモリーを制圧できる。

・パスワードがかかっている場合、attack（）の引数にパスワードを入力することで制圧する

制圧されたメモリーを取り戻す方法

・recovery （）を用いることで制圧されたメモリーを取り戻すことができる

・自身のメモリーからrecoveryを実行できない

・パスワードがかかっている場合、引数にパスワードを代入することで制圧が可能

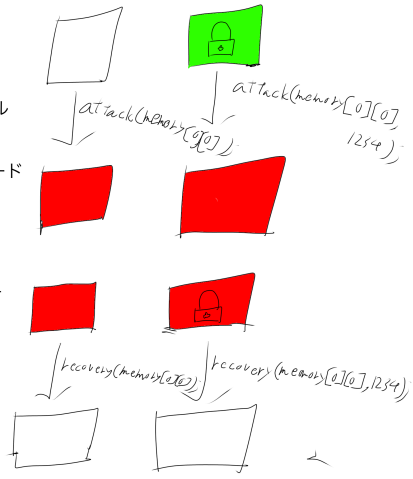

図４　制圧の仕方