

# 並列プログラミング Parallel Programming

2018 2Q

演習 第5回

情報理工学院 情報工学系

# 本日の流れ

- 課題内容の説明
- 演習に取り組む

## ● 目的

- java Stream の並列計算を体験
- java の動態の観察 jvusualvm

## ● 題材

- 映像の減色処理を行うプログラム
- 衝突判定を並列で行う図形表示プログラム
- ゲームプログラム

# 課題のダウンロード

次のどちらかからダウンロードしてください

学内アクセス

- **OCW-i**

- **[www.img.cs.titech.ac.jp/lecture/para/](http://www.img.cs.titech.ac.jp/lecture/para/)**

# 準備 (1)

- ダウンロードした para5.tar.gz を展開する

```
tar zxvf para5.tar.gz
```

```
tar xvf para5.tar
```

← ブラウザが勝手に一段階  
解凍した場合

- 解凍後のディレクトリ

## Para5(トップディレクトリ)

— Makefile (後述)

— README 説明文

— sweet.png

— **bin** クラスファイル (\*.class) が格納される (解凍直後は空)

— **javadoc** ドキュメントが格納される (解凍直後は空)

— **src** ソースファイル

— **lib** 実行に必要なjavaのライブラリ集(jarファイル)

## 準備 (2)

- ソースファイルを `javac` コマンドでコンパイルしてクラスファイルを作る
  - 今回の演習では **トップディレクトリ**で

Calculator が他のクラスに依存する場合、順次コンパイルしてくれる

```
javac -d bin -encoding UTF-8 -sourcepath src  
-cp bin src/para/calc/Calculator.java
```

実際は一行で書く

として下さい

<code>-d bin</code>	コンパイル後のクラスファイルをディレクトリ <code>bin</code> に置く
<code>-encoding UTF-8</code>	ソースファイルの文字コードが <code>UTF-8</code> であることを示す
<code>-sourcepath src</code>	ソースファイルがディレクトリ <code>src</code> 以下にあることをコンパイラに教える
<code>-classpath bin</code>	依存するクラスやライブラリのありかをコンパイラに教える

※ `man javac` としてその他のオプションを確認すること

# 準備 (3)

- java コマンドでクラスファイルを実行する
  - 今回の演習ではトップディレクトリにて

```
java -cp bin:lib/* para.Main04
```

パッケージ名 起点となるクラスの名前

として下さい (デモ用プログラムは para.Main?? と para.Game01 があります )

-cp bin:lib/\*

-cp は -classpathの短縮形

実行に必要なクラスファイルがディレクトリbin 以下に置かれていること、標準以外のjavaライブラリファイルがlib/に置かれているを java コマンドに教える

※ man java としてその他のオプションを確認すること

# 準備 (4)

- javadoc コマンドでソースファイルのコメント文からHTMLのドキュメントファイルをつくる

```
package para.calc;
import javafx.application.Application;
...
/** JavaFXで作成する電卓プログラム. */
public class Calculator extends Application
{
    /** 入力文字列表示領域. */
    Label input;
```

外部の Javadoc  
文書へのリンク

実際は一行で  
書く

今回の演習では、Para5 直下で、

```
javadoc -sourcepath src -charset utf-8 -encoding UTF-8
        -link https://docs.oracle.com/javase/jp/10/docs/api
        -d javadoc -package para.game para para.graphic.shape
para.graphic.target para.graphic.parser
para.graphic.camera
```

HTML文書の出力先  
ディレクトリ

パッケージ名

としてください

※ man javadoc としてその他のオプションを確認すること



# 準備 (5)

- コマンドをいちいちタイプするのが面倒 ...
  - 今回は Makefile を用意したので make コマンドで javac , javadoc の実行が簡単に行える

make Main09	Main09 をコンパイルして、実行
make Main10	Main10 をコンパイルして、実行
make Main11 PARAMETER=SINGLE	Main11 をコンパイルして、引数SINGLEを与えて実行
make Main11 PARAMETER=PARALLEL	Main11をコンパイルして、引数PARALLELを与えて実行
make Game01	Game01 をコンパイルして、実行
make clean	bin 以下のクラスファイルをすべて削除
make javadoc	javadoc コマンドを実行

ソースコードを更新してもmakeが感知しない場合があるので、コードを書き換えても結果に変化がない場合は、一度 make clean して再コンパイルして下さい

上を実行すると実際に発行されたコマンドが表示される  
Makefile を自分好みに変更してよいです

Makefileの記述ではタブ\tは意味があります。スペースで置き換えると、makeが正しく解釈できません。  
Makefileの書き方は各自調べて下さい

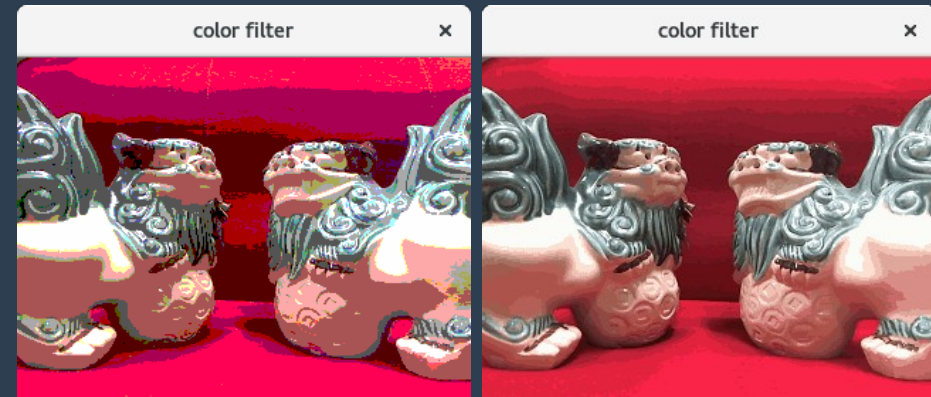
# 課題 0

- javadoc コマンドを実行して HTML 文書を生成し、ブラウザで閲覧する
  - Mac OS X では `open HTMLファイル名` とすればブラウザが起動する
  - 各クラスのパッケージ名などを確認する
  - ブラウザのエンコーディングの設定は UTF-8 にする
  - コンパイルエラーが起こる場合は展開直後にトップディレクトリで一度 `make javadoc` とタイプすると解決する場合がある

MacOSではダウンロード直後のファイルは安全を疑い、`._*`ファイルを作りアクセス制限をOSがする。その`._*`ファイルを消す作業が `make javadoc`には含まれている

# 課題 1

para.Main09とpara.Main10はカメラ映像を減色して表示するプログラムである。Main09の方が処理が単純で高速であるが、Main10の方が減色しても元の色との違いが少なく画質は良い。Main10の減色処理を並列化し、処理速度を改善せよ



1.1) 減色する前のカメラ画像では、個々の画素の色には表現可能色  色中の1色が用いられている。Main09とMain10では各画素はそれぞれ  色中と  色中の1色が用いられる。空欄に入る数を答えよ。

1.2) Main10が使用しているpara.TargetColorFilter2 のメソッドprocessの内部には、Streamを使った処理が3箇所ある。これらをParallelStreamに変換して並列化をすると並列化前とは異なった結果になってしまう。単純にParallelStreamに変換して問題となるStreamは何番目か。問題が発生するStreamが複数あると考える場合はそのを全て挙げよ。

1.3) 1.2での単純な並列化では並列化前と異なる結果を生じてしまう原因を説明せよ。

1.4) 1.3の原因を取り除き、全てのStreamの処理を並列化対応させて、ParallelStreamに変換し、Main10の高速化を実現せよ。変更した点を説明せよ。

## 課題 2

玉とブロックが衝突するとブロックは消えて、玉は完全弾性で跳ね返る運動を続けるプログラムpara.Main11を実行して並列処理が計算時間を短縮するために有効であることを確認し、さらに計算時間を短縮するための改善を施せ



2.1) para.Main11は100単位時間分の玉の運動を計算すると終了する。起動時に引数として「SINGLE」を与えると並列処理なしで演算を行い、「PARALLEL」を与えると並列処理ありで演算が行われる。出力される時間は何を計測した時間であるかを説明し、どの程度高速化されたかを数値で答えよ。なお、計測するにあたり、計測条件や計測方法として注意した点も答えよ。

**hint** マルチタスクOSでは他の起動中のプロセスにより計測時間にばらつきが生じる。

2.2) プログラムの一部分だけの演算時間を調べるためのソフトウェアであるjvisualvmを使って、より詳細に演算時間を調べよ。

演習室のiMacでは、ターミナル上で、  
/Library/Java/JavaVirtualMachines/jdk1.8.0\_172.jdk/Contents/Home/bin/jvisualvm  
とフルパスでコマンドファイルして起動させる。

(つづく)

## 課題 2

玉とブロックが衝突するとブロックは消えて、玉は完全弾性で跳ね返る運動を続けるプログラム `para.Main11` を実行して並列処理が計算時間を短縮するために有効であることを確認し、さらに計算時間を短縮するための改善を施せ



衝突判定部分は並列化により何倍速くなったかを `jvisualvm` を用いて調査せよ。  
`jvisualvm` をどのように利用したのかを説明し、数値を交えて高速化の調査結果を答えよ。

**hint** `jvisualvm` で得られる数値の意味を調べてから回答すること

2.3) 衝突判定部分は並列化による改善法が実装されているが、それ以外の部分でプログラム全体を高速化するにあたり、効果的な改善対象はどこであろうか。

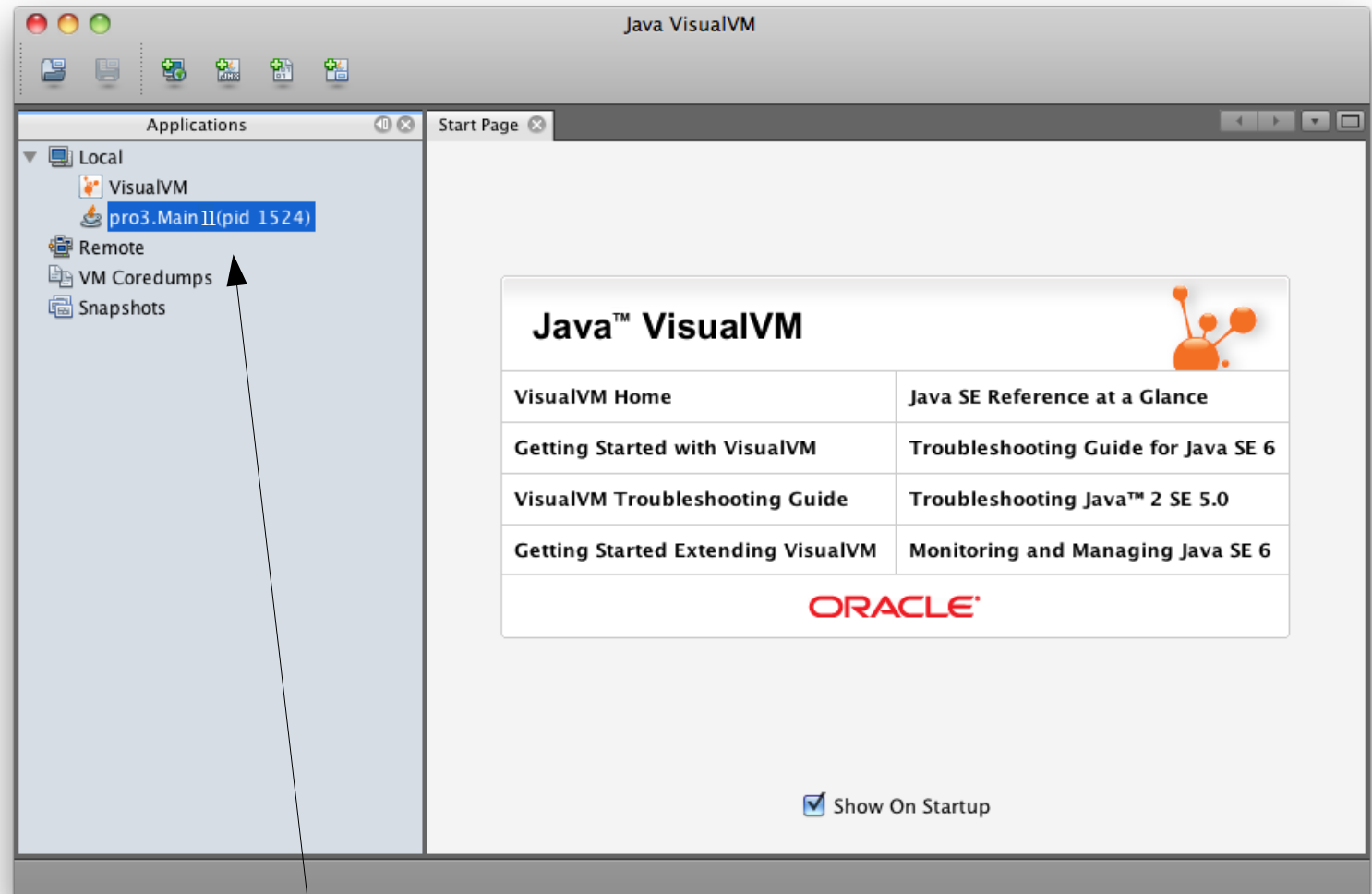
`jvisualvm` を用いて、改善対象を選び、その部分を高速にするようプログラムを改良せよ。改良部分のコードがどこであるかと、どのような方針で改良したかを説明せよ。

2.4) 2.3で改良した部分のみに関して、改良前と比べ、処理時間は何%になったかを `jvisualvm` を用いて調べて回答せよ。ただし処理時間の定義にはこの場合にもっとも適すると考えるものを選び、その定義も説明せよ。



## 課題 2 補足説明 jvisualvm

- Step1  
jvisualvmを起動
- Step2  
Main11を起動

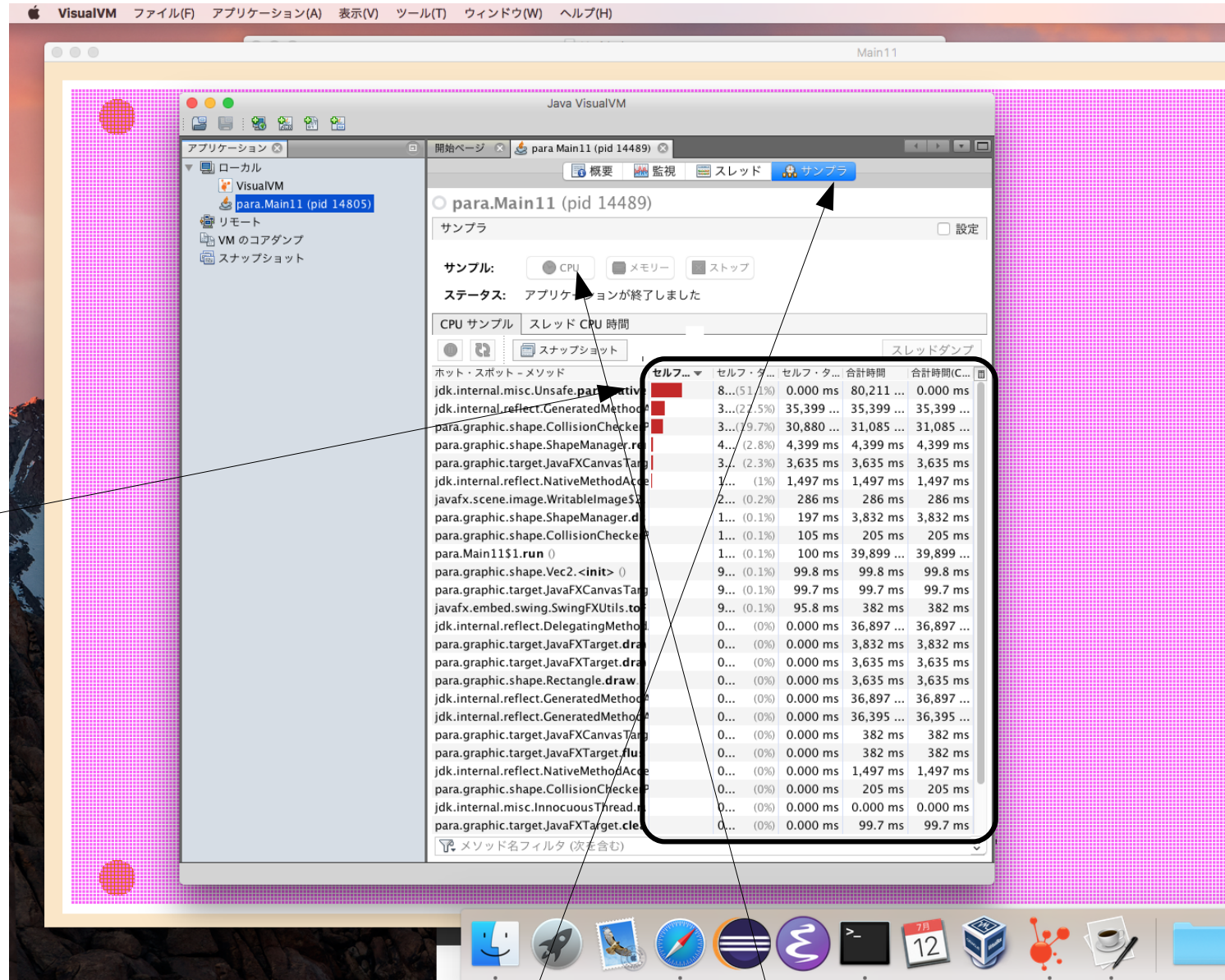


Main11 を走らせると、それを動かすjava VMが現れる  
ダブルクリックでjvisualvmとそれとの接続を開始する

## 課題 2 補足説明 jvisualvm

- Step1  
jvisualvmを起動
  - Step2  
Main11を起動
  - Step3  
Main11の観察開始
- 色々な測定値が表示される

色々な測定値が表示される



サンプラをClick してから、CPUをクリックして  
CPUサンプルを開始

## 課題 3

para.Game03は「ブロック崩し」ゲームの雛形である。楽しめるようにゲーム性を高めよ。

**ユーザの操作はキーボードのFキー（左）とJキー（右）**

3.1)ゲーム性を高めるため以下を実装せよ

- 得点ルールを定義しその得点を刻々表示する
- ゲームの終了条件を定める
- ゲームオーバーの後、スタートボタンを押すと新しくゲームを開始できる
- スタートボタンの左にある数値選択機能を利用して、ゲーム開始時に難易度を設定できるようにする

上記の実装部分がコード中のどこか分かるように説明せよ

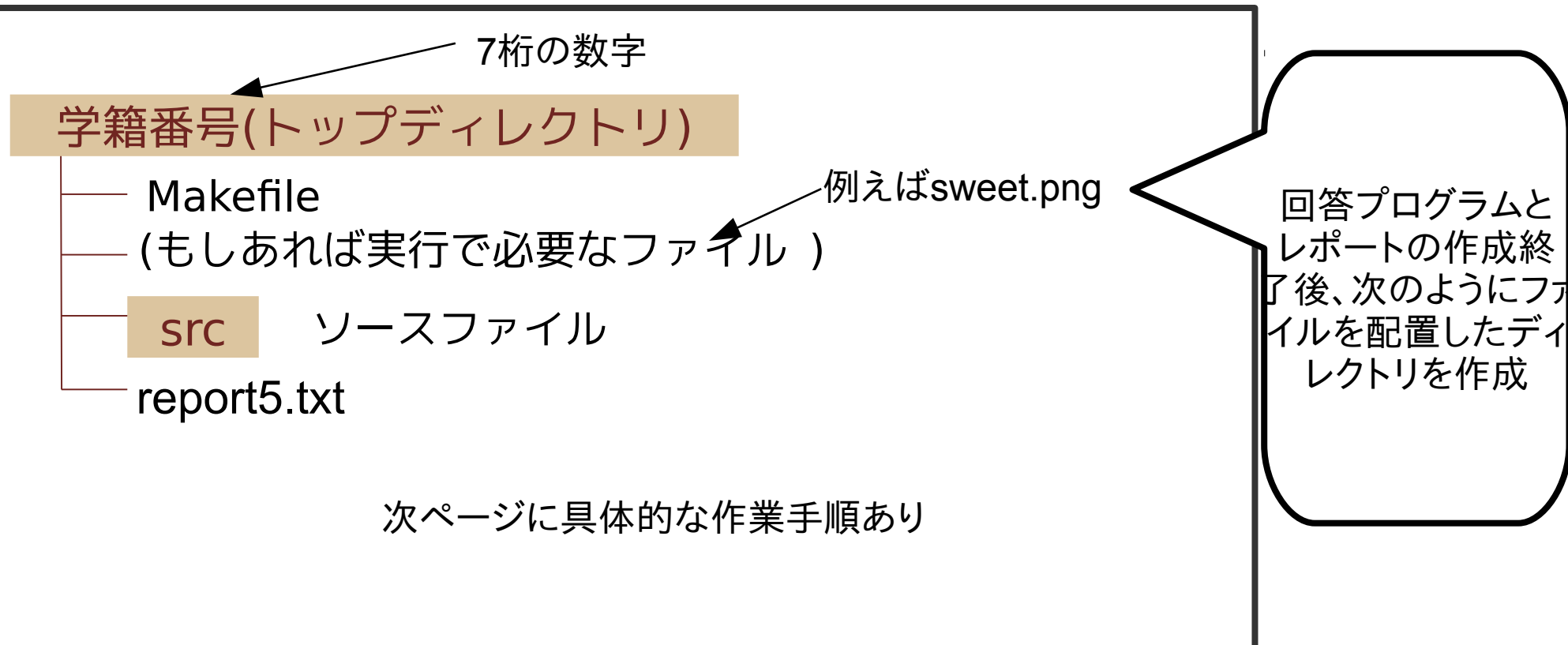
3.2)さらにゲーム性を高める方法を1つ以上考え、それを実装せよ

どのような方法であるかを説明し、その実装部分がコード中のどこか分かるように説明せよ



# 提出方法 (1 of 3)

- para5.tar.gz を展開したディレクトリ構造を保ったまま，課題の変更作業を行う
- 各課題で自分が変更したファイルの先頭には自分の名前と学籍番号を書いておく
  - プログラムの場合はコメント内に書く
- 課題 1 から 3 の回答文、工夫点および感想を書いた report5.txt を用意する（雛形は課題のウェブページ）



# 提出方法 (2 of 3)

- 提出用ディレクトリを作成する 学籍番号から7桁の数字にすること

```
mkdir dir
```

- ソースファイルのディレクトリのコピーを作る

```
cp -R トップディレクトリ/src dir
```

今回はPara3

- dir* に Makefile report5.txt もコピーする

```
cp トップディレクトリ/Makefile トップディレクトリ/report5.txt dir
```

- dir* に課題3で必要なその他のデータがあればコピーする

```
cp トップディレクトリ/otherfiles dir
```

例えばsweet.png

- 次のコマンドを実行する

```
zip ex5-1612345.zip -r dir
```

学籍番号に対応する7桁の数字にすること

*dir* 以下の内容が圧縮され、ex5-1612345.zip が作られます

- 圧縮後に内容を “unzip ex5--1612345.zip” で確認すると提出ミスを防げて安全

# 提出方法 (3 of 3)

- 作成した zip ファイルを ocw にアップロードする
- 締め切り
  - 7月20日(金) 10:00am (JST)

# 質問したいときは ...

- メールアドレス

pro3report [AT] img.cs.titech.ac.jp

※ [AT] は @ に置き換え

– 佐藤、齋藤、担当 TA へメールが送られます

- あるいは西 8 号館 E 棟 401 号室の佐藤まで  
直接質問に来て下さい

– 4F エレベータホールのインターホンにて  
呼び出し