知能情報工学実験演習II
C++演習 (Part 2)

グループ A+B：下薗

グループ C+D：中村

ＴＡ：原田、井上

グループ演習

- 1グループ：5〜6人
  - グループ単位でプログラムを作成
- 対象となるゲームは 大富豪／大貧民
- 演習課題
  - 知的な?戦略を持つコンピュータプレイヤーを実装する
- テキスト
  ~sin/DAIHUGOU2013/doc.pdf
- このスライド
  ~sin/DAIHUGOU2013/slide.pdf

大富豪/大貧民：仕様

- 52枚のカードとジョーカー1枚を使用、強さは
  ジョーカー＞2＞A＞K＞Q＞J＞10＞...＞3 とする.

- ターンのまわってきたプレイヤーは、場に出ているカード（リード）より強いカードを出すか、パスをする.

- 全員パスして一巡したら、最後にカードを出したプレイヤーが新たなリードを手札から出す.

- カードがなくなったプレイヤーは上がる.
  最初に上がったプレイヤーが大富豪となる.

- リードは同じ数字の 2 ペア、3 カード、4 カードでもよい.
  カードを出す（フォロー）には、同じ枚数でより強いカードを出さなければならない.

※ シークエンス組、革命、都落ち、搾取交換、その他のローカルルールは考えない.

サンプルプログラム

- サンプルプログラム

  ~sin/DAIHUGOU2013

  - Card.{h|cc}
  - CardSet.{h|cc}
    - 基本的には前回までと同じものだが，若干の拡張があるのでこちらを使うこと
  - Dealer.{h|cc}
    - ゲームの管理を行う
  - Player.{h|cc}
    - 基本となるプレイヤー
  - LittleThinkPlayer.{h|cc}
    - 拡張プレイヤーのサンプル
    - 実際には Player のサブクラス
  - ThinkTA1.{h|o}
    - 歴代 TA 伝統の作成思考ルーチン.
      オブジェクトファイルのみ提供し、ソースコードは秘密.
  - main.cc

C++ Card.cc CardSet.cc Dealer.cc Player.cc LittleThinkPlayer.cc ThinkTA1.o main.cc

プログラムのイメージ

- テキストのみ１対１のチャットでカードゲームをすることを想像する
  - Skype, MS Messenger で「部屋」はなし
  - ただし、通信は使わず分散開発でやります
- 複数が集まる「場」は、ディーラー（立会人）が各プレーヤーとチャットし実現
- 他のプレーヤーが「人間？」と誤解するようなものをめざせ？

API 概略： Dealer.{h|cc}

- ゲームを管理するクラス
  - プレイヤーとゲームの管理をする
  - カードの配布、場に出されたカードが受理できるかの判定など
  - 人間が遊ぶ場合とは異なり，プレイヤーではない
- 参考のためソースコードを提供するが、変数や関数は変更せず作成すること

API 概略： Player.{h|cc}

- プレイヤーのクラス
  - 手持ちのカードとプレイヤー名
  - 重要なメンバ関数は
    - bool follow(CardSet &, CardSet &);
      - カードを出す
    - bool approve(CardSet &, int[]);
      - ターンのプレーヤーが場に何を出したか確認する

    ※ 使用例は LittleThinkPlayer のソースコードを参照
  - この2つ以外の関数は変更しない
    - follow() および approve() で使用する関数を作るのはご自由に

API 概略： Player::follow()

- 引数： 場のカード pile 、出すカード s
  - もっともシンプルな「ランダムにカードを引いて出す」プレイヤー

```
bool Player::follow(CardSet & pile, CardSet & s) {
        Card tmp;
        s.makeempty();
        hand.pickup(&tmp, -1); // choose any card from my hand
        s.insert(tmp);
        return true;
}
```

- Card や CardSet のメンバ関数を使いなさい
  - カードの大小関係比較：bool isGreaterThan(Card c)
  - 等しいかどうか：bool equal(Card tgt)
  - ジョーカーかどうか：bool isJoker()

API 概略： Player::approve()

- 他人の出したカードを確認する
  - 場のカードには何もしないが
- 引数で受け取った pile には，現在の場に出ているカードが渡される
  - これを使って，他のプレーヤーが出したカードを確認できる
- 引数で渡される numCards には、現在の各プレーヤの手札数が格納されている
- LittleThinkPlayer を参考にしてください
  - memoryには、場に出されたカードを記録
  - 基本的には「他人の出した」カード
    - 正確には「場で通った」カード
  - 必要に応じ、変数を作成して保持枚数を管理するなどして戦略に使う

カードの判定(1)

- ゲームの進行、管理
  - Dealer クラスで行う．考えなくてよい
  - 判定は Dealer::accept() で行う
    - 受理されれば，場にカードを出して true
    - 受理されなければ，カードは手持ちに戻り false
  - 複数枚組（Jkr を含んでも可）のチェックも実装されている

カードの判定(2)

- ゲームとしての判定
  - 空のセット（０枚）、通らない（リードより弱い）カードセットを出すと、「パスします」と解釈
    - 通らなかったカードは自分の手持ちに戻る
  - 出すカードが通るかどうかは、当然、自分で判定した方が強いものができる

プログラムのイメージ

プレーヤーにそれぞれのターンを知らせる
そのプレイヤーの番に場に出ているカードを教える

Dealer

ゲームを公平に管理
カードの判定
順位の判定

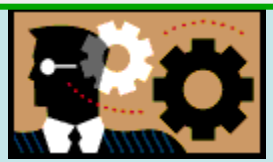
Player

follow()
approve()

follow()の戦略に基づきカードを選ぶ

follow()
approve()

受理できるかできないかを判定し,
受理できればカードを受け取り,
受理できなければプレイヤーに返す



follow()
approve()

approve()で誰かが場に出したカードをチェック

グループでの作業の進め方

- 協力して計画的にすすめなさい
- 大きな流れ（サイクル）
  - プレイヤー作成→対戦→議論→改良→....
- 例えば,
  - 最初は1人1つずつ作る
  - 対戦させ，問題点を考察する
  - いくつかのプレイヤーを統合して改良
  - このプロセスをくりかえし、
    最終的に班で1つの最強プレイヤーを準備

プレイヤーの作り方

- 1回のサイクルであまり欲張らなくていい
  - 「3人寄れば文殊の知恵」
  - 少しずつ賢くする：
    できるだけ弱いカードから出す、
    選んだカードが場に通るか考えて出す、
    複数カードを出せるようにする（かなり強力）
- follow() と approve() 以外を変更しないこと
  - 最初に提供したソースコードでコンパイル、リンク
  - 使用するのは Card と CardSet

複数プレイヤーの実現

- 継承によりサブクラスを作成すること
  - たとえば，LittleThinkPlayer.{h|cc} を参照

```
LittleThinkPlayer::LittleThinkPlayer(const char * s) : Player(s) {}

bool LittleThinkPlayer::approve(CardSet & pile, int numCards[]) {
        memory.insert(pile); // LTPのメンバ変数memoryにカードを格納
        // numCard[]には各プレイヤーの残り枚数が格納されている
}

bool LittleThinkPlayer::follow(CardSet & pile, CardSet & s) {
        Card tmp;
        s.makeempty();
        inHand().pickup(&tmp, -1); // anyway, choose a card.
        // Player の Private 変数には直接アクセスできないので注意
        s.insert(tmp);
        return true;
}

---main関数で
d.regist(new Player(“Erika”)); // 通常の player.cc の follow や approve
d.regist(new LittleThinkPlayer("Warabi")); // 通常の LittleThinkPlayer.cc の follow や approve
```

演習の進め方

- 一人の力に頼らない
  - みんなで分担すること
  - 必ず全員がプログラムを作成すること
    - 採用されたかどうかは問わない

- グループで作成したサブクラスが、
  他の班のプログラムと一緒に動くように注意
  - 班同士の対戦をするため
  - ソースファイル，クラス名は
    A 班ならば GroupA とすること

レポートについて(1)

- 班で1つ提出.
  - 表紙には，演習名，班員の学生番号と名前，提出日
  - 各班員の役割分担
    - 例えば，各繰り返しの段階で実装した戦略や作業などを箇条書きする程度でよい
      - ただし，誰が何をやったかはわかるようにすること
        - 最終的に採用されたかどうかは問わない
  - プログラムリスト
    - 最終的に実装した部分のみでよい
  - 実装した思考ルーチンの戦略など
    - 工夫した点、未完成の部分などを含む
  - 実行結果
    - 主要な部分・レポートでの説明に必要な部分のみでよい
  - 全体の考察とまとめ

レポートについて(2)

- 個人評価表を提出
  - 自己評価と班員の評価
  - A4用紙で1枚

- 提出の締切
  - 演習の終了した日の翌週の金曜17時まで
    ※その日が休業日の場合、その直近の業務日
  - 受け取り：　東棟６階に会場を設置予定

評価について

- 班のレポート＋個人評価票＋順位に基づく配点
- 最終プログラムを提出
  - Group[番号].cc および .h の提出
    - LittleThinkPlayer.{cc|h}のような形で
    - デバッグ用のコメントなどは出さないように！

- 最終回の15:30（予定）
  - 抽選により3班で1リーグの1次予選
    - ただし，各リーグともThinkTA1とベースのPlayerが参加
    - 順番を入れ替えて5回試行
    - コンパイルできなかったら，即敗退
  - 各リーグ1位が決勝進出
    - ただし，ThinkTA1が1名参加
    - 順番を入れ替えて5回試行

- 順位と得点
  - 1位：5点，2位：4点，3位：3点