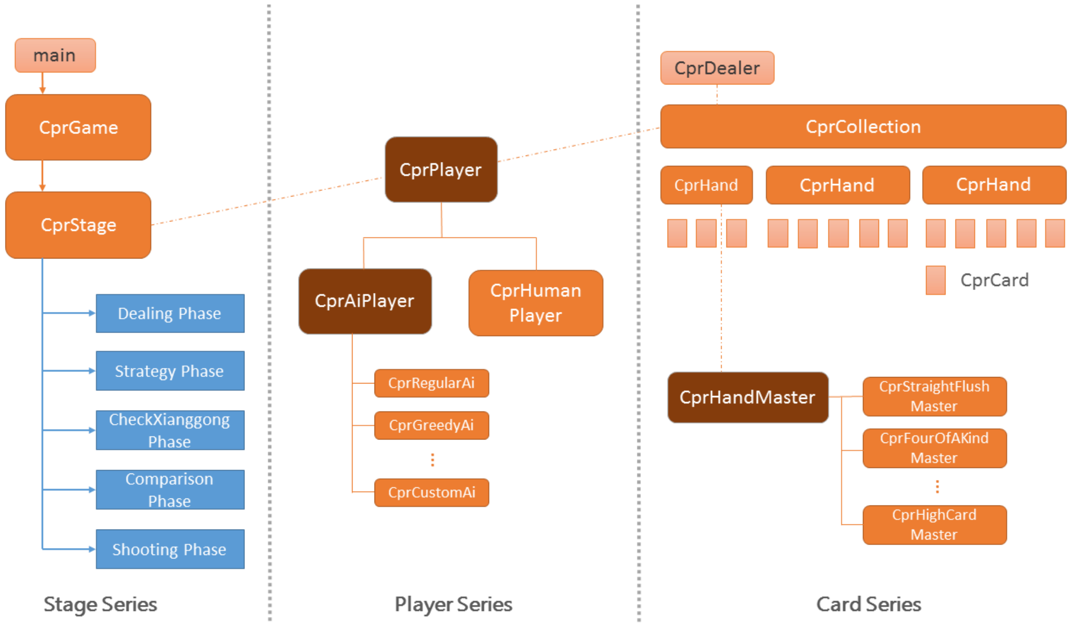
Computer Programming – Week 15 Practice

2016.12.21

這次有提供整體遊戲的範例執行檔(ChinesePoker_ref.exe，在 UNIX shell 下編譯的同學可以用 ChinesePoker_ref.out)，請大家熟悉遊戲的指令、玩法和規則。

Practice 1: Cpr*Master (Children of CprHandMaster)



For each HandType, there's a corresponding **Cpr*Master**. For example, **CprFullHouseMaster** masters in all tasks related to FullHouse.

The member functions of a Cpr[HTYPE]Master are as follows:

1-1 Description:

1. `bool determineMe(CprHand& hand);`

Given a hand, determine if it's of type HTYPE. If it's so, please **sort its cards by the order specified in 1-4** and then return true; otherwise, simply return false.

Note that it might be a 5-cards hand or 3-cards hand.

You can assume hand is NOT of a type stronger than HTYPE.

E.g.

```
CprFullHouseMaster fhouseMaster;
CprHand h1 = ...  // suppose h1._cards is {3d 5c 3c 5s 5h}
```

```
bool b = fhouseMaster.determineMe(h1);
  // now b == true, and h1._cards is {5c 5h 5s 3c 3d}
```

2. bool compareDeeply(const CprHand&, const CprHand&);

Assume that two CprHand arguments are both of the same type HTYPE, and has been sorted by the specified order (that is, they've been passed to Cpr[HTYPE]Master::determineMe previously.)

Compare these two hands. If the former is NOT stronger than(<=) the latter, return true; otherwise, return false.

E.g.

```
CprFullHouseMaster fhouseMaster;
CprHand h1 = ...  // suppose h1._cards is {5c 5h 5s 3c 3d}
CprHand h2 = ...  // suppose h1._cards is {ac ad as 2h 2s}
bool b = fhouseMaster.compareDeeply(h1, h2);  // true
```

3. bool containsMe(const CprCollection& col, CprHand& res);

Find if there's a hand of type HTYPE in a collection.
Modify res by the **largest possible** hand in col (if there' multiple, pick one with the smallest suit.)

E.g.

```
CprFullHouseMaster fhouseMaster;
CprHand col = ...
  // suppose col._availCards is { Ac 5c 2d 6d 5h 6h 2s 6s }
CprHand hand;
bool b = fhouseMaster.containsMe(col, hand);
  // now b == true
  // hand == {6d, 6h, 6s, 5c, 5h} (order not matter)
```

4. bool containsMeUnder(const CprCollection& col, const CprHand& uppHand, CprHand& res);

[Optional] *Not necessary to implement if you're not ambitious to create a strong AI.*

Find if there's a hand LESS than uppHand of type HTYPE in a collection.

Modify res by the **largest possible** hand less than(<) uppHand in col. (if there' multiple, pick one with the smallest suit.)

E.g.

```
CprFullHouseMaster fhouseMaster;
CprHand col = ...
  // suppose col._availCards is { Ac 5c 2d 6d 5h 6h 2s 6s }
CprHand hand;
CprHand upperHand = ...
  // suppose upperHand._cards is { 6d 6h 6s 4c 4s}
bool b = fhouseMaster.containsMeUnder(col, upperHand, hand);
  // now b == true
  // hand == {6d, 6h, 6s, 2d, 2s} (order not matter)
```

1-2 Tasks in/after coding session:

We're going to implement CprFullHouseMaster, CprStraightMaster and CprOnePairMaster this week. Select ONE different from your group members; that is, all of these three are being implemented by some person in your group.

Please take a look at **CprEachMaster.h**. Since each Cpr*Master has identical members, we use a macro to define them.

The implementation of CprHighCardMaster is provided to you as a reference, try if you can successfully compile&run test_HighCard.cpp. Furthermore, you can use test_src/test_*.cpp to test your implementation.

In order to implement Cpr*Master::determineMe, you might want to do some similar things for **CprHand** as for CprCollection in practice 1.

1-3 Homework: Check Point #1 (due by Jan. 4 21:00)

Please implement ALL Cpr*Master classes (group-based), and test the correctness of them by yourselves. (we only test some simple cases on judgegirl; that is, you still need to self-test your masters carefully.)

You're recommended to implement each **compareDeeply** and **determineMe** first, and containsMe later. (In your final released project, fail to correctly implement each compareDeeply and determineMe will be regarded as FATAL ERROR; however, fail to correctly implement containsMe and containsMeUnder just makes your AI stupid.)

Uploading your codes onto judgegirl and FTP before Jan. 4 21:00. (Judge Girl 測試後，請將每個小孩 Cpr*Master.cpp 上傳至 FTP)

Please also provide a readme.docx to demonstrate your progress (for example, whether you've finished all the tasks, or whether you'd finished each optional Cpr*Master::containsMeUnder, and so on.各個 Cpr*Master.cpp 完成的進度)

1-4 HandTypes:

Please refer to https://en.wikipedia.org/wiki/List_of_poker_hands and the following information:

The sorted example indicates how should a hand be sorted after it's passed as an argument to Cpr*Master::determineMe.

Note that in Chinese Poker, **suit is NOT being compared.** (only rank matters)

HandType	說明
StraightFlush	10JQKA > 910JQK > … > 23456 > A2345 不同花同點的同花順等大 Sorted example: {Ac, 2c, 3c, 4c, 5c}, {2c, 3c, 4c, 5c, 6c}, {10c Jc Qc Kc Ac}
FourOfAKind	先比同點的四張牌，一樣則比第五張 Sorted example: {7c 7d 7h 7s 3c}
FullHouse	先比同點的三張牌，一樣則比對子 Sorted example: {7c 7d 7s 4c 4s}
Flush	比最大牌，如一樣大則比次大者，依此類推 Sorted example: {Ah Qh 7h 3h 2h}
Straight	10JQKA > 910JQK > … > 23456 > A2345 三張牌不能成為順子。 Sorted example: {Ac, 2d, 3c, 4c, 5c}, {2c, 3d, 4c, 5c, 6c}, {10c Jd Qc Kc Ac}
ThreeOfAKind	先比同點的三張牌，一樣則比大單張，再一樣則比小單張 Sorted example: {7c 7d 7s Ac 4s}, {8d 8h 8s}
TwoPairs	先比較大的對子，如相同再比第二對，又相同則比單張 Sorted example: {Qc Qd 7c 7s 10s}, {Ad As 2c 2s 5d}
OnePair	先比對子，相同則比最大單張，再相同則比次大單張，依此類推 Sorted example: {Qc Qd As Ks 4c}, {Qc Qd Ad}
HighCard	比最大牌，如一樣大則比次大者，依此類推 Sorted example: {As Ks Qd 10h 4c}, {Qc 10d 4d}

Practice 2:   CprHandMaster

As the parent of Cpr*Master, CprHandMaster is an **abstract** class. To deal with any task related to the “Card Series”, it’s members are defined as follows:

2-1   static member functions:

1. **`static void DetermineType(CprHand& hand);`**

   Determine the type of hand, modify its _type, and sort it by preferred order.

   Note: HandType is an **enum** defined in CprHandType.h.
   A good reference about enum can be found here.

   E.g.

```
CprHand h1 = ...  // suppose h1._cards is {3d 5c 3c 5s 5h}
CprHandMaster::DetermineType(h1);
  // now h1._cards is {5c 5h 5s 3c 3d},
  // and h1._type is HTFullHouse
```

2. **`static bool Compare(const CprHand&, const CprHand&);`**

   Compare two hands (whose types have been determined), and return true if the former is not stronger than the latter; and vice versa.

   E.g.

```
CprHand h1 = ...  // suppose h1 is {5c 5h 5s 3c 3d}, HTFullHouse
CprHand h2 = ...  // suppose h2 is {3s 4s 5d 6d 7h}, HTStraight
bool b = fhouseMaster.compareDeeply(h1, h2);  // false
```

3. static CprHand GetBestHand(const CprCollection& col);

Find the best possible hand in col and return it.

E.g.

```
CprHand col = ...
  // suppose col._availCards is { Ac 5c 2d 6d 5h 6h 2s 6s }
CprHand hand = CprHandMaster::GetBestHand(col);
  // now hand._cards is {6d, 6h, 6s, 5c, 5h}
```

4. static CprHand GetBestHandUnder(const CprCollection&, const CprHand& upperHand);

Find the best hand **weaker** than upperHand in col and return it.

E.g.

```
CprHand col = ...
  // suppose col._availCards is { Ac 5c 2d 6d 5h 6h 2s 6s }
CprHand upperHand = ...
  // suppose upperHand._cards is { 6d 6h 6s 4c 4s}
CprHand hand = CprHandMaster::GetBestHandUnder(col, upperHand)
  // now hand._cards is {6d, 6h, 6s, 2d, 2s}
```

5. static CprHandMaster* CreateMaster(HandType type);

Allocate a new Cpr[type]Master and return a CprHandMaster* pointing to it.

6. static string TranslateType(HandType);

To translate HandType to human readable string.

2-2 virtual functions:

1. `virtual bool compareDeeply(const CprHand&, const CprHand&);`
2. `virtual bool determineMe(CprHand&);`
3. `virtual bool containsMe(const CprCollection&, CprHand&);`
4. `virtual bool containsMeUnder(const CprCollection&, const CprHand&, CprHand&, bool = false);`

2-3 Tasks in coding session:

Please finish **CprHandMaster::Compare** and **CprHandMaster::GetBestHand**. Note that **CprHandMaster::DetermineType** has been done for you as a reference.

CprHandMaster::CreateMaster and CprHandMaster::TranslateType are also done. However, once you finish a new Cpr*Master, you need to uncomment the corresponding lines in those two functions.