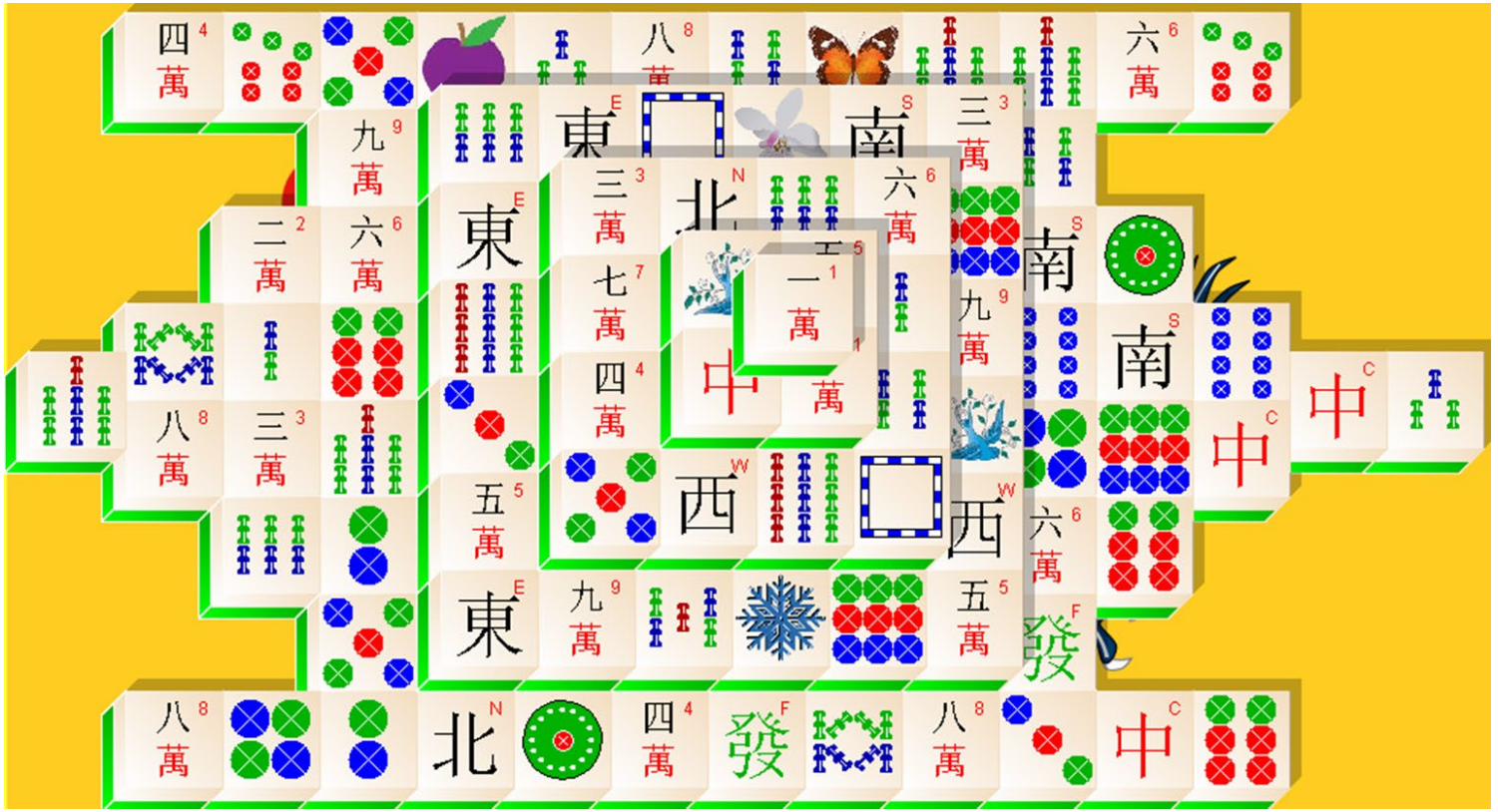
MAHJONG TILES

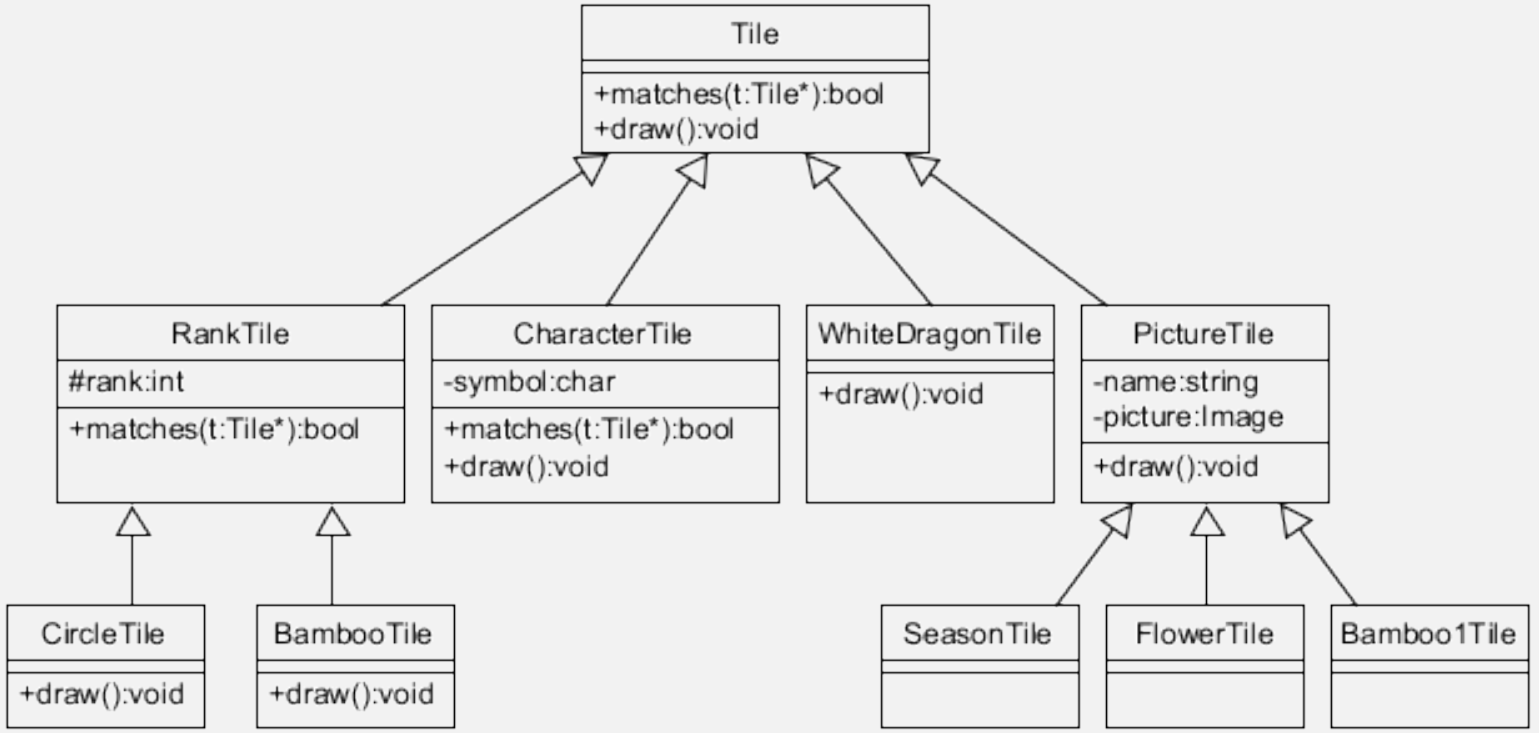
Outlining A Polymorphic Solution

Delroy A. Brinkerhoff

MAHJONG STACKED TILES



TILE CLASS DIAGRAM



MATCHING TILES

```
Tile* t1 = first selected tile;
Tile* t2 = second selected tile;

if (t1->matches(t2))
{
    remove(t1);
    remove(t2);
}
```

- t1 ≠ t2
- t2 can't be null
- Must be instances of the same class
- Rank tiles must be the same rank
- Character tiles must have the same symbol

Tile MATCHES

```
class Tile
{
    public:
        virtual bool matches(Tile* t)
        {
            if (this == t)
                return false;

            if (t == nullptr)
                return false;

            return typeid(*this) == typeid(*t);
        }
};
```

LOGICAL-AND AND SHORT-CIRCUIT EVALUATION REVIEW

left	right	Left && right
F	F	F
F	T	F
T	F	F
T	T	T

- Operands are evaluated left to right
- Short-circuit evaluation stops the evaluation when the result is determined

RankTile MATCHES

```
class RankTile : public Tile
{
    private:
        int rank;

    public:
        RankTile(int r) : rank(r) {}

        virtual bool matches(Tile* t)
        {
            return Tile::matches(t) && rank == ((RankTile *)t)->rank;
        }
};
```

CharacterTile MATCHES

```
class CharacterTile : public Tile
{
    private:
        char symbol;

    public:
        CharacterTile(char c) : symbol(c) {}

        virtual bool matches(Tile* t)
        {
            return Tile::matches(t) && symbol == ((CharacterTile *)t)->symbol;
        }
};
```