Chess Program - Part 4 - Make Chess Object Oriented

Modify the `Chess` program so that it uses an Object-Oriented approach (OOP) using a *class* object rather than being Procedure Oriented (POP).

1. Keep your constants "as is" as `static const` at the top, don't try to integrate them into the class. Constants never change and are fine as external declarations.
2. Add a method called `play()` that wraps the functionality that used to be in `main()`
3. `main()` should now be only,

```
int main()
{
      Chess chessGame = Chess();
      chessGame.play();
      return 0;
}
```

4. You should have three private variables `mRow`, `mColumn`, and `mType`. Note that as `mRow`, `mColumn`, and `mType` are now a private variables, they no longer need to be passed into `drawBoard()` and similar, as done previously.
5. Below is the class definition for the `Chess` class you should create.
6. Use Preprocessor commands to break this program into a three files, `main.cpp`, `chess.hpp`, and `chess.cpp`. Keep your constants at the top of `chess.hpp`
7. Add a `Chess()` constructor that prints when a `Chess` instance is created, cout << "Chess object being created. " << this << endl;
8. and a `~Chess()` destructor that prints when a `Chess` instance is destroyed, cout << "Chess object being deleted. " << this << endl;

The structure of your OOP program should look like so,

```
static const int BOARD_SIZE = 8;

// pieces
static const char ROOK = 'r';
static const char KNIGHT = 'n';
static const char BISHOP = 'b';
static const char QUEEN = 'q';
static const char KING = 'k';


class Chess
{
public:
    void play();
    void drawBoard();

private:
    int mRow;
    int mColumn;
    char mType;
```

```
    void askInt();
    void askType();
    bool isValidType(char type);
};
```