Плюсы:

1. Проект организован в виде классов;
2. Код выдержан в едином стиле написания:

```
// блюдо (не тарелка, а элемент меню)
class Dish : public QObject
{
	Q_OBJECT
	private:
		QString name;	// название
		float price;	// цена
		int quantity, quantityToCook;	// количество (и какой партией готовить)
		int popularity; // популярность
		QList<bool> requiredUtensils;  // требуемые приборы
	public:
		Dish(QString _name, float _price, int _quantity, int _popularity, QList<bool> _reqUtensils, QObject *parent = 0);
		// методы получения свойств
		float getPrice();
		int getQuantity();
		int getQuantityToCook();
		int getPopularity();
		QString getName();
		QList<bool> getReqUtensils();
		// замена данных
		void replaceData(QString _name, float _price, int _quantity, int _popularity, QList<bool> _reqUtensils);
		bool take();  // взять
	signals:
		needToCookMore();
	public slots:
		void cook();  // приготовить
};
```

3. Наименование классов и полей «говорящие» (см. скриншот выше);
4. Информативные комментарии к интерфейсу классов;

Минусы:

1. Есть методы большого размера, которые можно поделить на более мелкие;
2. Низкая информативность комментариев, а также их отсутствие в реализациях методов;
3. Использование непонятных констант:

```
0		for (int i = 0; i < 4; i++)
1		{
2			if (freeSides[i])
```

4. Присутствуют закомментированные участки кода:

```
/*
void Model::cookDishes()
{
	needToCookMore = true;
}
*/
```

5. Отсутствует перенос строк в некоторых местах, из-за чего приходится использовать прокрутку;

Субъективная оценка:

4.5 из 10.