**Добрый день, 16а группа!**

Продолжаем общаться дистанционно.

Сегодня мы познакомимся со средой программирования Visual Studio

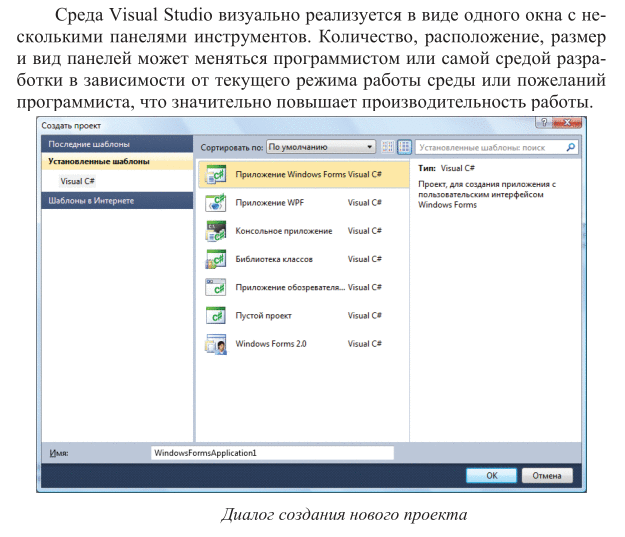
Задать вопросы, а также прислать ответы вы можете

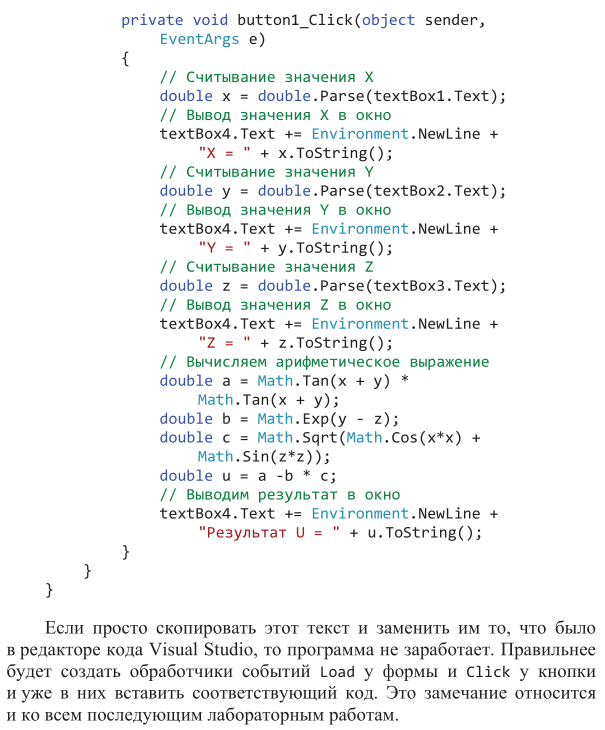
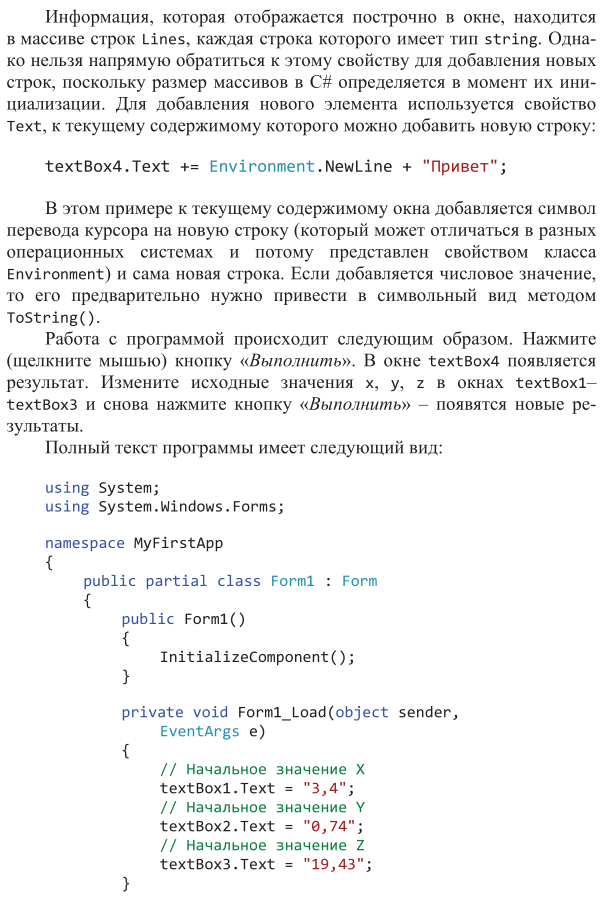
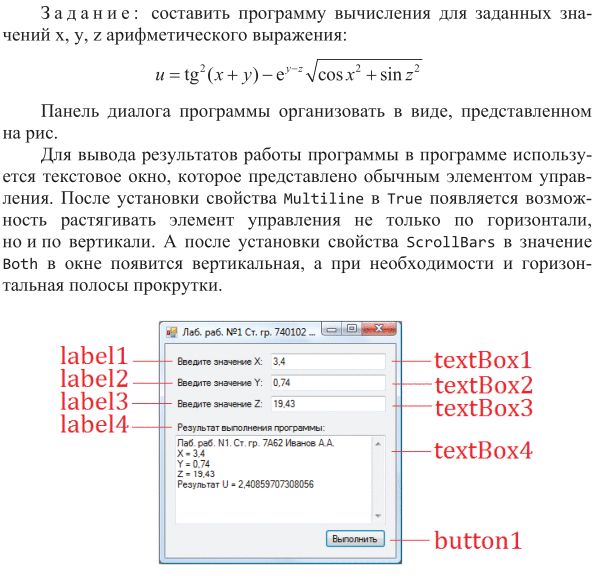
1. на адрес электронной почты: [ddrmx@ya.ru](mailto:ddrmx@ya.ru)
2. через соцсеть <https://vk.com/ddrmx>

С уважением, Максим Андреевич.

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ ПО ТЕМЕ:

Программа. (1 ЧАС)





ЗАНЯТИЕ ПО ТЕМЕ:

Компьютерные модели различных процессов (MVC). (1 ЧАС)

MVC — это шаблон программирования, который позволяет разделить логику приложения на три части:

* **Model (модель)**. Получает данные от контроллера, выполняет необходимые операции и передаёт их в вид.
* **View (вид или представление)**. Получает данные от модели и выводит их для пользователя.
* **Controller (контроллер)**. Обрабатывает действия пользователя, проверяет полученные данные и передаёт их модели.

Может показаться, что это что-то запутанное, но на самом деле всё просто.

## Как работает MVC

Лучше всего понять концепцию MVC можно на реальном примере — ресторане с фастфудом. В нём посетители (пользователи) подходят к кассиру (одновременно вид и контроллер), видят меню и заказывают какое-нибудь блюдо.

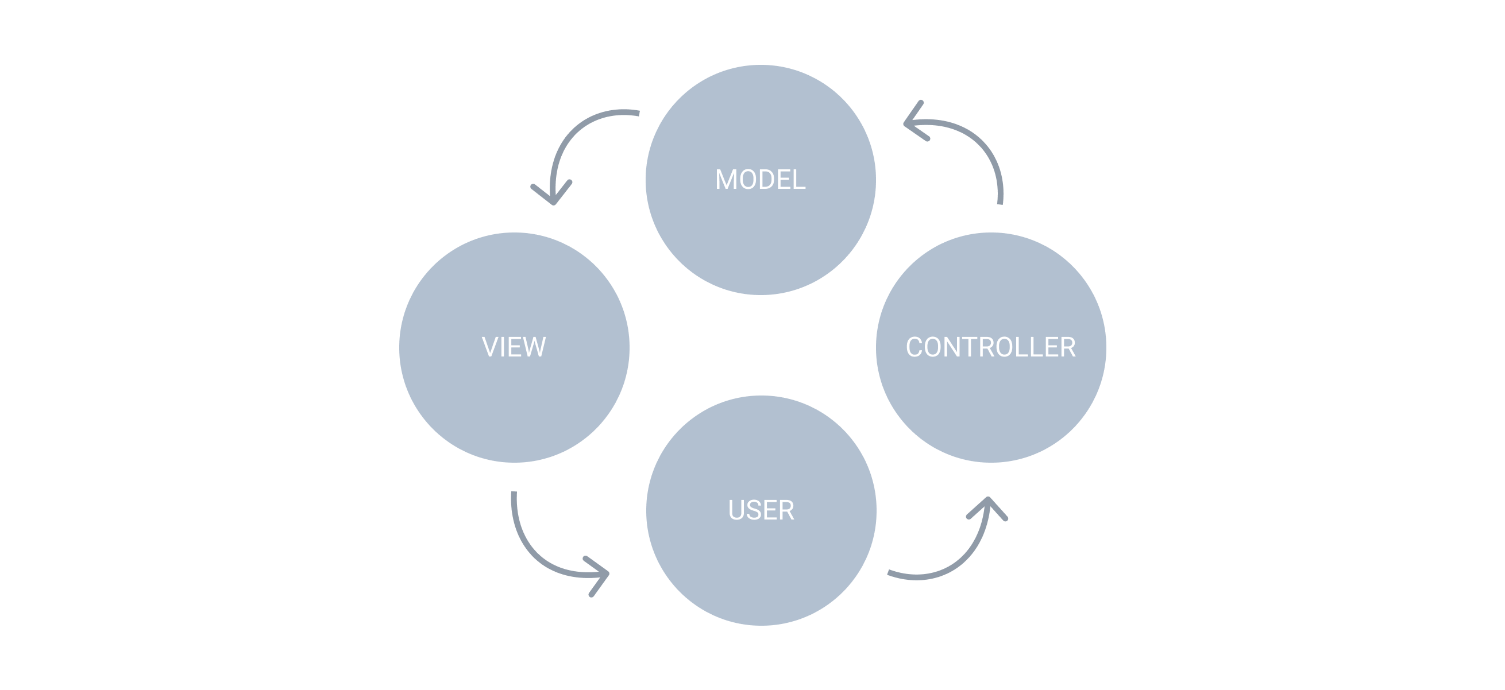
Кассир проверяет, всё ли в порядке с заказом, и после оплаты передаёт нужные данные повару (модель). Повар готовит заказанное блюдо, хотя понятия не имеет о том, как выглядит посетитель, оплатил ли он заказ и так далее.

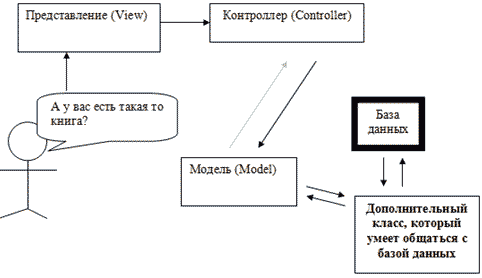
Когда модель закончит свою работу, она отправит результат в вид — обратно кассиру, который, в свою очередь, отдаст готовое блюдо посетителю.

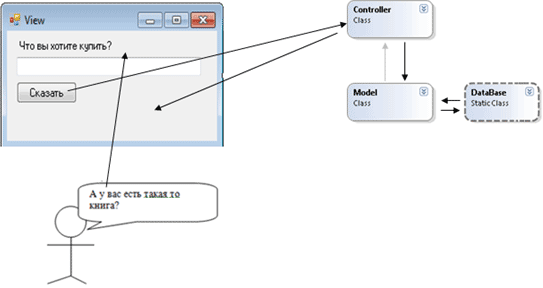
Если же говорить о приложениях, то компоненты будут следующие:

* **Вид** — интерфейс.
* **Контроллер** — обработчик событий, инициируемых пользователем (нажатие на кнопку, переход по ссылке, отправка формы).
* **Модель** — метод, который запускается обработчиком и выполняет все основные операции (получение записей из базы данных, проведение вычислений).

Стоит также отметить, что реализация паттерна MVC может отличаться в зависимости от задачи. Например, в веб-разработке модель и вид взаимодействуют друг с другом через контроллер (как в примере с рестораном), а в приложениях модель может сама уведомлять вид, что нужно что-то изменить.







## Письменно ответьте на вопрос: Зачем программистам нужен MVC?