**Практическое занятие №6**

**Создание горизонтальных и вертикальных меню**

**Цель:** Научиться создавать, с использованием css, горизонтальные и вертикальные меню

**Состояния ссылок**

Состояние **a:hover** - мы поймаем состояние, когда на ссылку навели курсор мышки. В этот момент мы можем, к примеру, поменять цвет ссылки или убрать/добавить ей подчеркивание.

Конструкция **:hover** называется ***псевдоклассом***. Псевдоклассы позволяют отлавливать разные состояния элементов.

Кроме **:hover** если еще псевдоклассы **:link**, которые отлавливают не посещенную ссылку, и **:visited**, которые отлавливают посещенную ссылку.

Есть еще и псевдокласс **:active**, который отлавливает следующее состояние: *на элемент нажали мышкой, но еще не отпустили*.

В следующем примере для ссылки в состоянии **:hover** убирается подчеркивание, в состоянии **:link** будет красный цвет, в состоянии **:visited** - зеленый, в **:active** - голубой:

a:link {

color: red;

}

a:visited {

color: green;

}

a:hover {

text-decoration: none;

}

a:active {

color: blue;

}

<a href="#">Ссылка</a>

В начале ссылка будет красного цвета, после нажатия на нее - зеленого, если нажать на нее мышкой и не отпускать - голубого, а если навести мышкой - станет неподчеркнутой.

Псевдоклассы наследуют друг от друга. К примеру, если я уберу подчеркивание для состояния **:link**, то оно уберется для всех состояний.

Из-за наследования для корректной работы данные псевдоклассы следует размещать именно в таком порядке, как в примере: *:link, :visited, :hover, :active* (ненужные можно не писать). Этот порядок подчиняется следующему мнемоническому правилу: **LoVe HAte**.

Часто состояния **:link** и **:visited** объединяют вместе через запятую:

a:link, a:visited {

color: red;

}

a:hover {

text-decoration: none;

}

a:active {

color: blue;

}

В таком случае можно их вообще и не указывать:

a {

color: red;

}

a:hover {

text-decoration: none;

}

a:active {

color: blue;

}

## Сложные селекторы с учетом состояний ссылок

Наверняка на сайте у вас будут ссылки разных видов и, чтобы отличить их друг от друга, вы будете давать им разные классы или ложить в блоки с определенным id.

Пусть у нас есть ссылки с классом **.test** и без него. Выберем только ссылки с этим классом:

<a href="#">Ссылка без класса</a>

<a class="test" href="#">Ссылка с классом test</a>

<a class="test" href="#">Ссылка с классом test</a>

<a class="test" href="#">Ссылка с классом test</a>

a:link.test, a:visited.test {

color: red;

}

a:hover.test {

color: blue;

}

Пусть у нас есть ссылки внутри блока с id **test**. Выберем только ссылки только из этого блока:

<a href="#">Ссылка вне блока</a>

<article id="test">

<a href="#">Ссылка внутри блока</a>

<a href="#">Ссылка внутри блока</a>

<a href="#">Ссылка внутри блока</a>

</article>

#test a:link, #test a:visited {

color: red;

}

#test a:hover {

color: blue;

}

## Блок 1. Работа с границами на CSS

Сейчас мы с вами научимся добавлять **границу** элементам. Это делается при помощи трех свойств: свойство **border-width** задает толщину границы, **border-color** - цвет, а **border-style** задает тип границы.

Первые два свойства работают очевидным образом: **border-color** принимает цвета в том же формате, что и свойство **color**, а толщина границы может задаваться в любых единицах измерения (кроме процентов), чаще всего в пикселях.

А вот свойство **border-style** может принимать одно из нескольких значений: **solid** (сплошная линия), **dotted** (граница в виде точек), **dashed** (граница в виде черточек), **ridge** (выпуклая граница), **double** (двойная граница), **groove** (вогнутая граница), **inset** (рамка), **outset** (рамка) или **none** (отменяет границу).

Сделаем, к примеру, границу толщиной 3 пикселя, в виде точек, красного цвета:

article {

border-width: 3px; /\* толщина 3px \*/

border-style: dotted; /\* в виде точек \*/

border-color: red; /\* красный цвет \*/

width: 300px;

height: 100px;

}

Давайте теперь посмотрим на примерах, как выглядят различные типы границы.

### Значение solid

Значение **solid** - сплошная линия:

article {

border-width: 1px;

border-style: solid;

border-color: black;

width: 300px;

height: 100px;

}

### Значение dotted

Значение **dotted** - линия в виде точек:

article {

border-width: 1px;

border-style: dotted;

border-color: black;

width: 300px;

height: 100px;

}

### Значение dashed

Значение **dashed** - линия в виде тире:

article {

border-width: 1px;

border-style: dashed;

border-color: black;

width: 300px;

height: 100px;

}

### Значение ridge

Значение **ridge** - выпуклая линия:

article {

border-width: 3px;

border-style: ridge;

border-color: black;

width: 300px;

height: 100px;

}

### Значение double

Значение **double** - двойная линия:

article {

border-width: 3px;

border-style: double;

border-color: black;

width: 300px;

height: 100px;

}

### Значение groove

Значение **groove** - вогнутая линия:

article {

border-width: 3px;

border-style: groove;

border-color: black;

width: 300px;

height: 100px;

}

### Значение inset

Значение **inset** - рамка:

article {

border-width: 3px;

border-style: inset;

border-color: black;

width: 300px;

height: 100px;

}

### Значение outset

Значение **outset** - рамка:

article {

border-width: 3px;

border-style: outset;

border-color: black;

width: 300px;

height: 100px;

}

## Блок 2. Свойство-сокращение для границ

Так же, как и для шрифтов, для границ тоже существует **свойство-сокращение border**, которое мы можем использовать вместо того, чтобы писать 3 разных свойства для **толщины**, **цвета** и **типа** границы. В свойстве **border** мы можем просто перечислить эти значения, их порядок при этом не важен. Смотрите пример:

<article></article>

article {

width: 300px;

height: 100px;

border: 1px solid red;

}

Чаще всего гораздо удобнее пользоваться свойством-сокращением, чем отдельными свойствами.

## Блок 3. Граница для отдельных сторон

Существуют также свойства-сокращения для отдельных сторон: **border-left** (левая граница), **border-right** (правая граница), **border-top** (верхняя граница), **border-bottom** (нижняя граница).

Давайте сделаем блоку **только левую границу** с помощью свойства **border-left**:

<article></article>

article {

width: 300px;

height: 100px;

border-left: 1px solid red;

}

А теперь одновременно сделаем и **левую**, и **правую** границы:

<article></article>

article {

width: 300px;

height: 100px;

border-left: 1px solid red;

border-right: 1px solid red;

}

## Блок 4. Скругленные уголки

Сейчас мы с вами научимся **скруглять уголки** у границ. Это избавит наши сайты от некоторой угловатости и придаст им плавность линий.

Уголки границ (и фона, который мы разберем ниже) скругляются свойством **border-radius**, которое принимает значение в **пикселях** или **процентах** (или других единицах CSS).

Давайте скруглим уголки блоку, задав ему скругление в **10px**:

<article></article>

article {

width: 300px;

height: 100px;

border: 1px solid red;

border-radius: 10px;

}

Что означает то, что мы указали скругление в **10px**? Это **радиус круга**, который можно вписать в это скругление. Если у вас нелады с математикой и вам не понятно последнее предложение - забудьте о нем и просто подбирайте скругление на глаз. При некотором опыте это сделать не проблема (измерительного инструмента для измерения скруглений не существует, по крайней мере я о таком не слышал).

Имейте ввиду, что **border-radius** не входит в свойство-сокращение **border**, это разные свойства, хоть и имеют похожие названия.

## Блок 5. Разные скругления для разных углов

Сейчас мы с вами научимся делать разные скругления для разных углов. Как это сделать: свойство **border-radius** может не только одно значение, но и **два**, **три** или **четыре**. Каждое значение будет задавать скругление для своего угла. Давайте посмотрим более подробно.

### Четыре значения

Свойство **border-radius** может принимать 4 значения. Как в этом случае будут скругляться уголки - смотрите на следующем примере: **border-radius: 10px 20px 30px 40px** - **10px** - верхний левый угол, **20px** - верхний правый угол, **30px** - нижний правый угол, **40px** - нижний левый угол.

Смотрите, что у нас получится:

<article></article>

article {

width: 300px;

height: 100px;

border: 1px solid red;

border-radius: 10px 20px 30px 40px;

}

### Два значения

Если же в **border-radius** написать два значения, то первое задаст скругление углов одной диагонали, а второе - другой.

Смотрите на примере:

<article></article>

article {

width: 300px;

height: 100px;

border: 1px solid red;

border-radius: 10px 40px;

}

### Три значения

Ну, и наконец, если в **border-radius** написать три значения, то **первое** задаст скругление верхнего левого угла, **третье** - нижнего правого, а **второе** - скругление углов диагонали (двух оставшихся углов).

Смотрите на примере:

<article></article>

article {

width: 300px;

height: 100px;

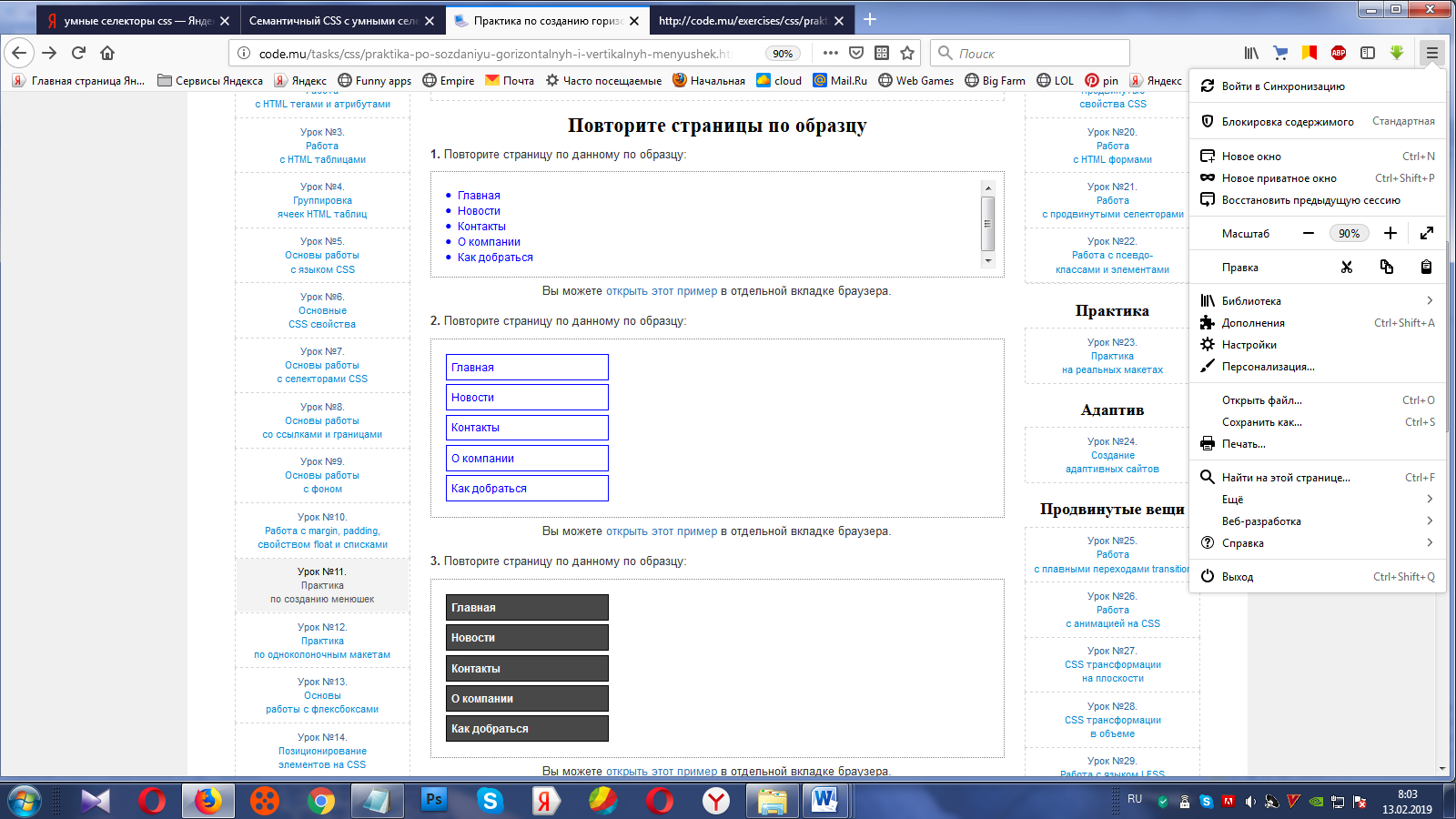
border: 1px solid red;

border-radius: 10px 20px 30px;

}

**Повторите страницы по образцу**

1. Повторите страницу по данному по образцу:



ul{

padding: 0px;

margin: 0px;

}

.clearfix{

clear: both;

}

.menu1 {

color: blue;

list-style-position: inside;}

.menu1 a:link, .menu1 a:visited{

text-decoration: none;

color: blue;

font: 13px Arial;

}

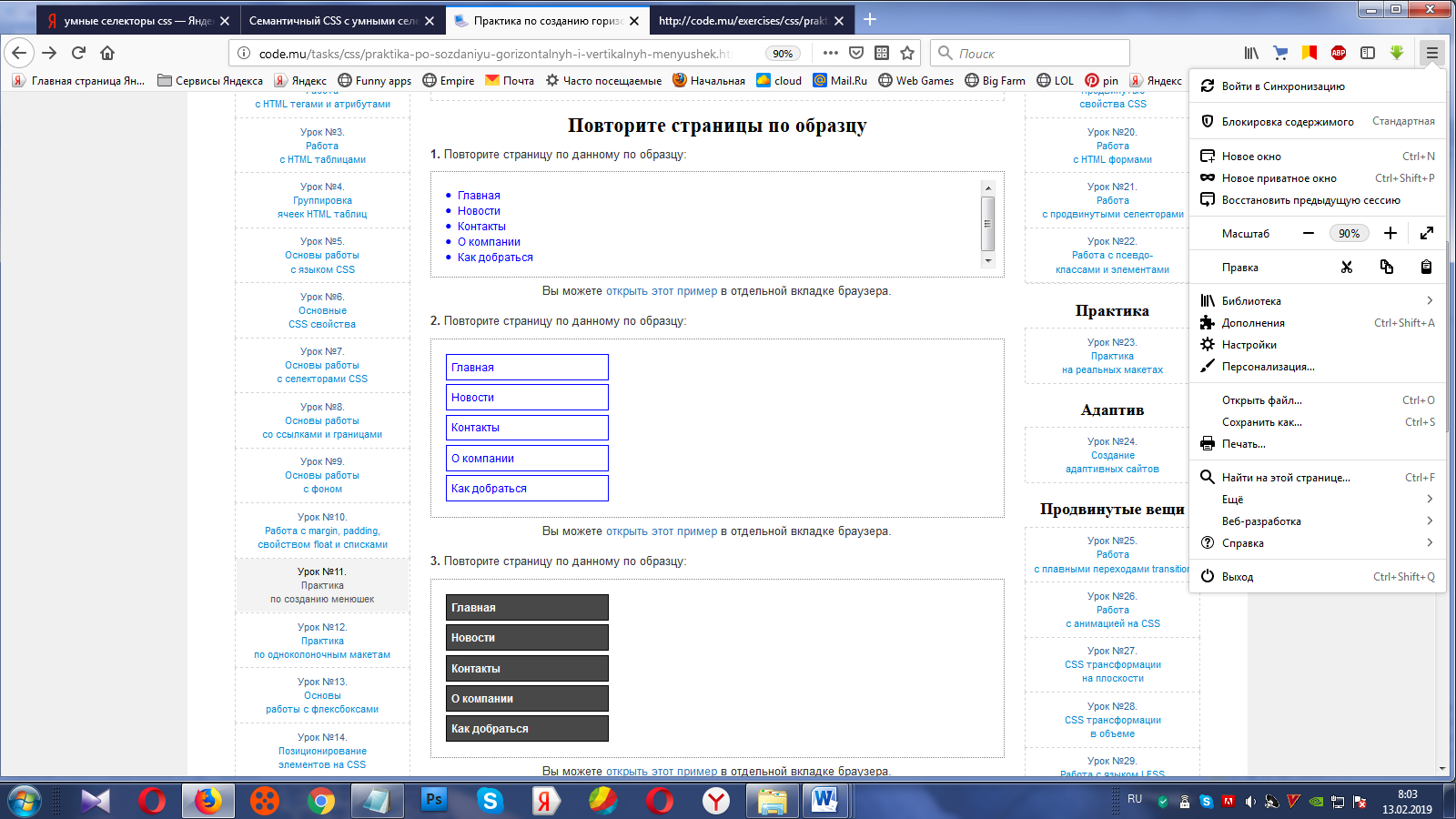
.menu1 a:hover{

text-decoration: underline;

color: red;

}

2. Повторите страницу по данному по образцу:



.menu2 {list-style-type: none; width: 200px}

.menu2 li{margin-bottom: 5px;}

.menu2 a:link, .menu2 a:visited{

text-decoration: none;

color: blue;

font: 13px Arial;

border: 1px solid blue;

display: block;

padding: 0px 5px;

line-height: 30px;

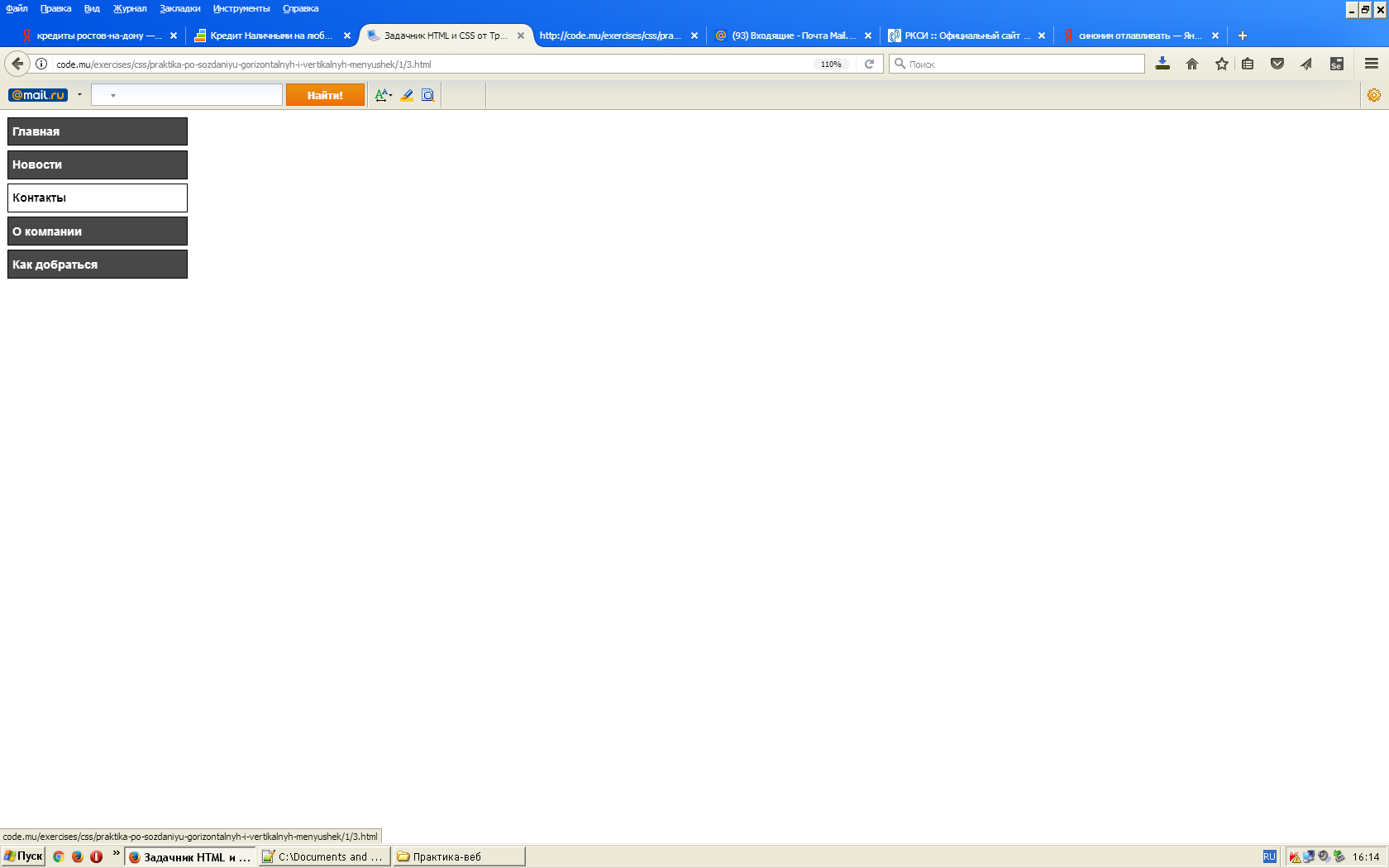
}

.menu2 a:hover{

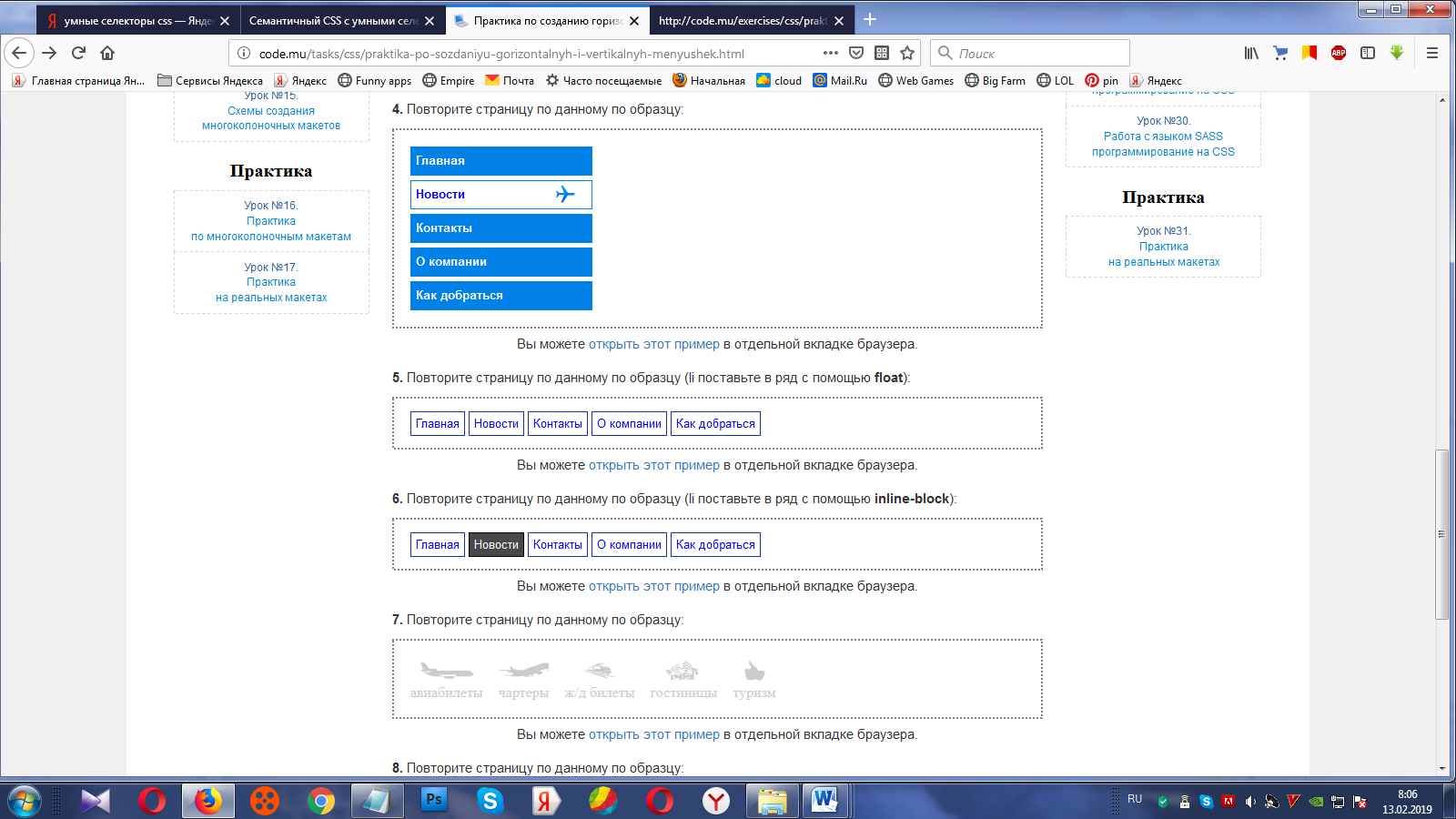
font-weight: bold

}

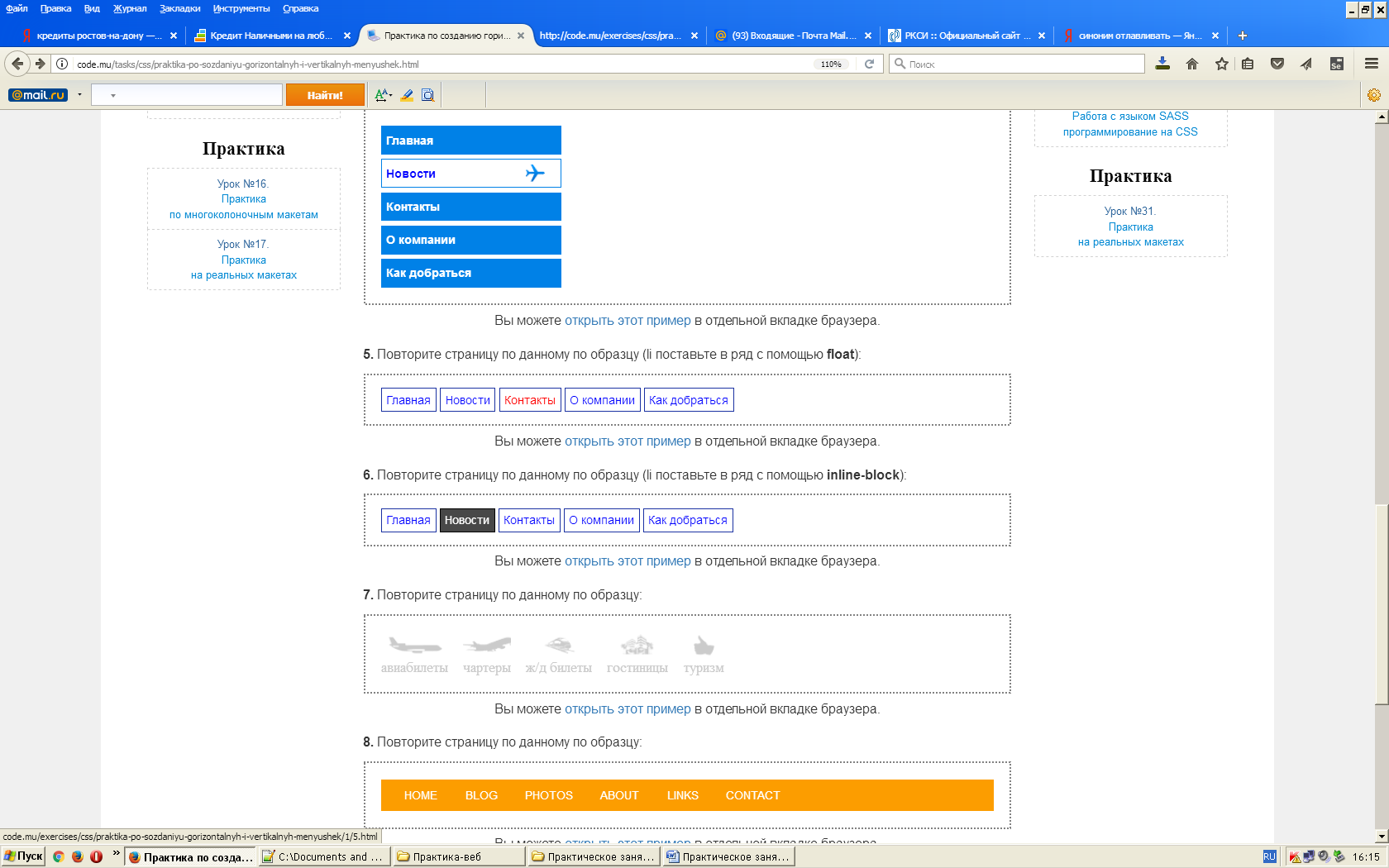
3. Повторите страницу по данному по образцу:



4. Повторите страницу по данному по образцу:



5. Повторите страницу по данному по образцу:



.menu5 {list-style-type: none;}

.menu5 li{margin-right: 4px; float: left;}

.menu5 a:link, .menu5 a:visited{

text-decoration: none;

color: blue;

font: 13px Arial;

border: 1px solid #1A31A2;

padding: 5px;

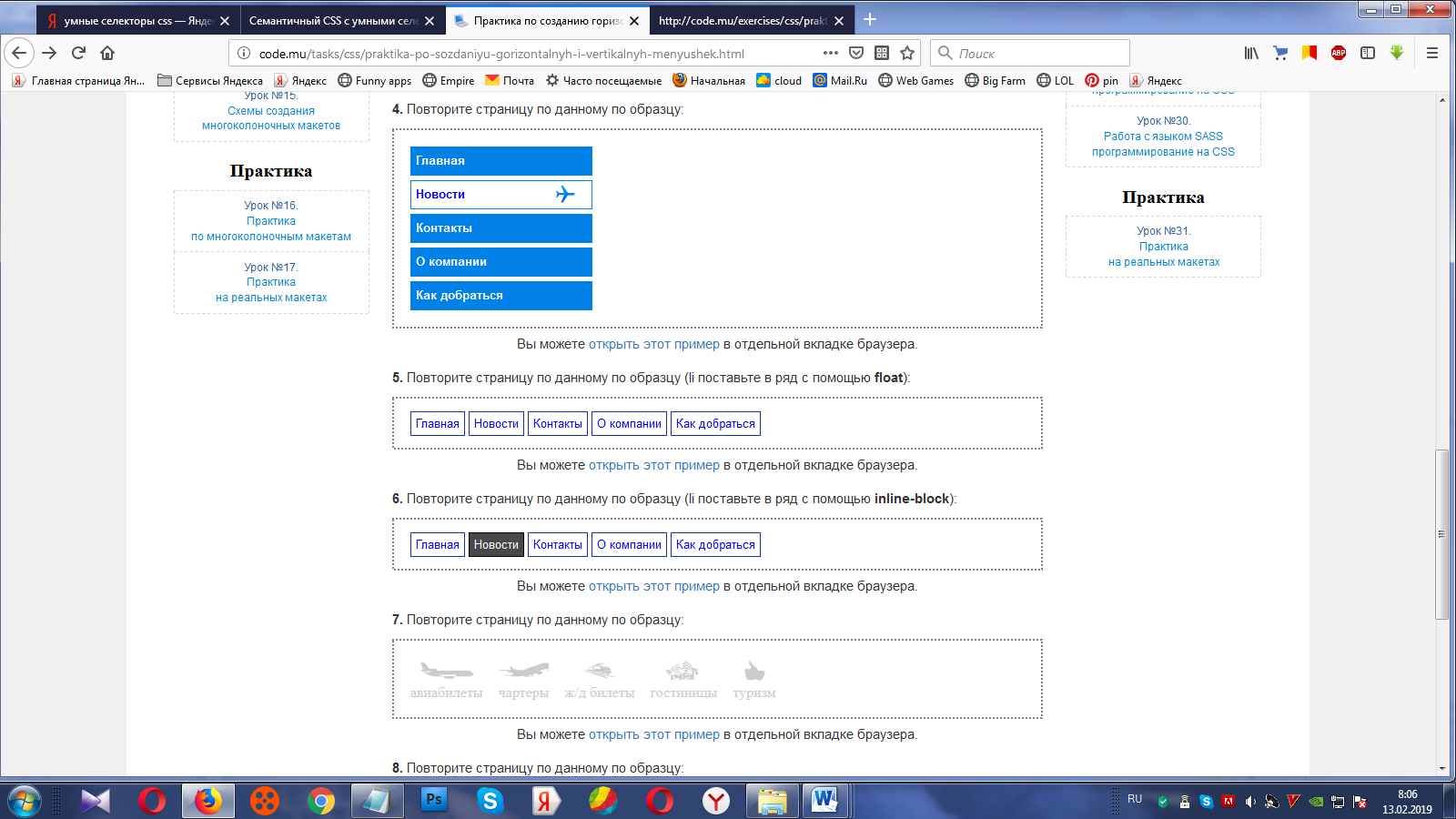
}

.menu5 a:hover, .menu5 a.active{

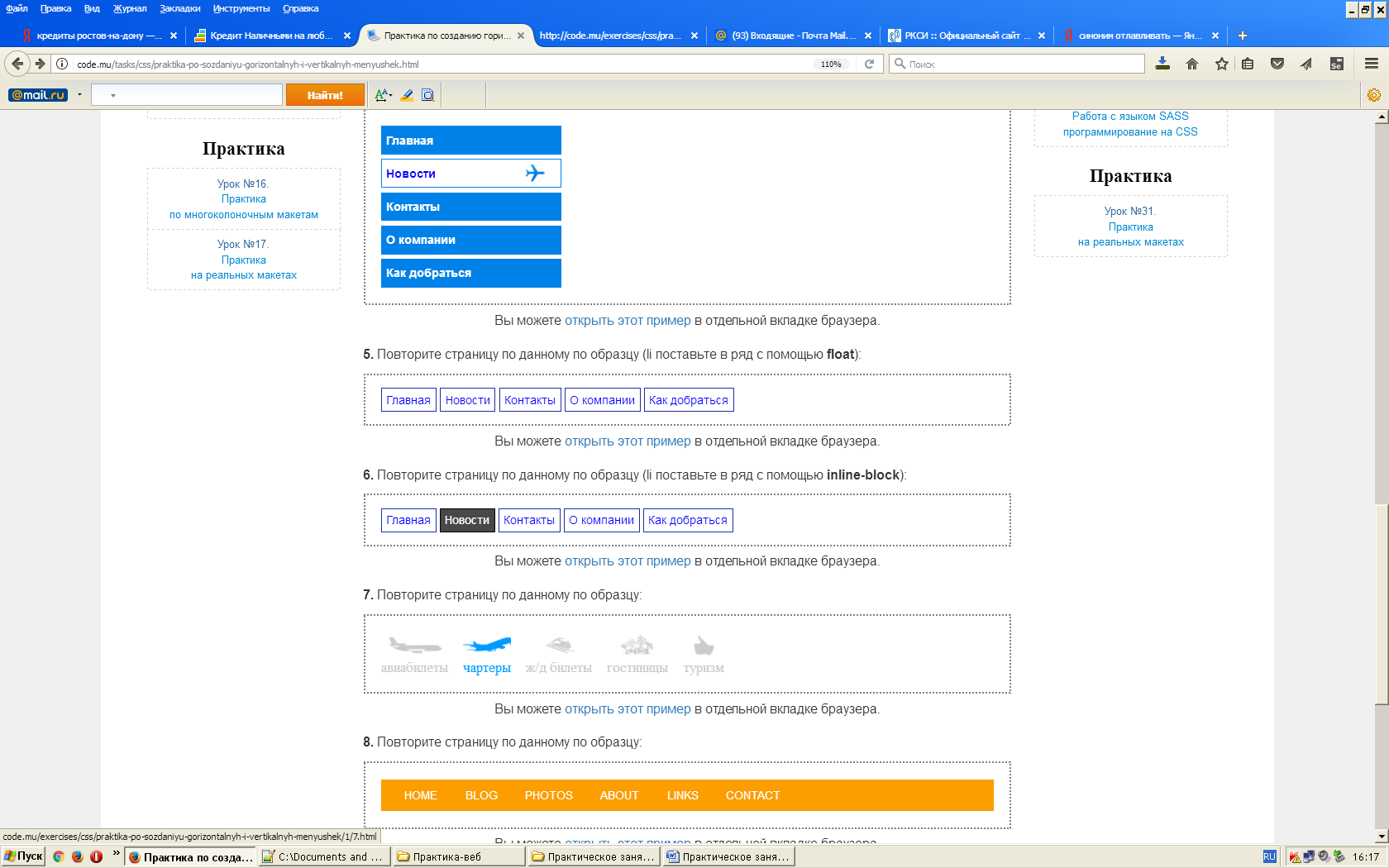
color: red;

}

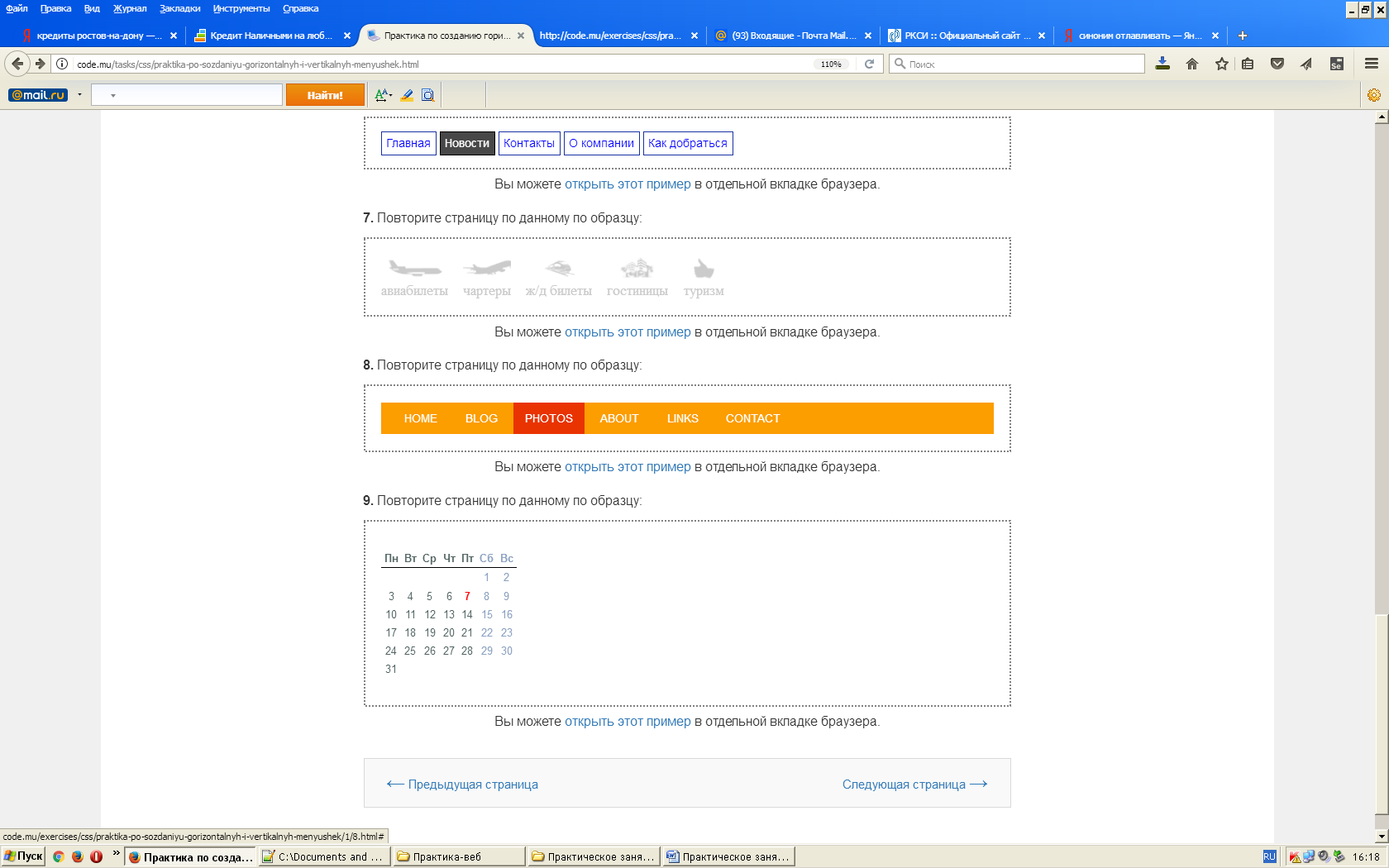
6. Повторите страницу по данному по образцу:



7. Повторите страницу по данному по образцу:



8. Повторите страницу по данному по образцу:



9. Повторите страницу по данному по образцу:

