Градиент, фон, фильтр

Данильченко Анна Александровна

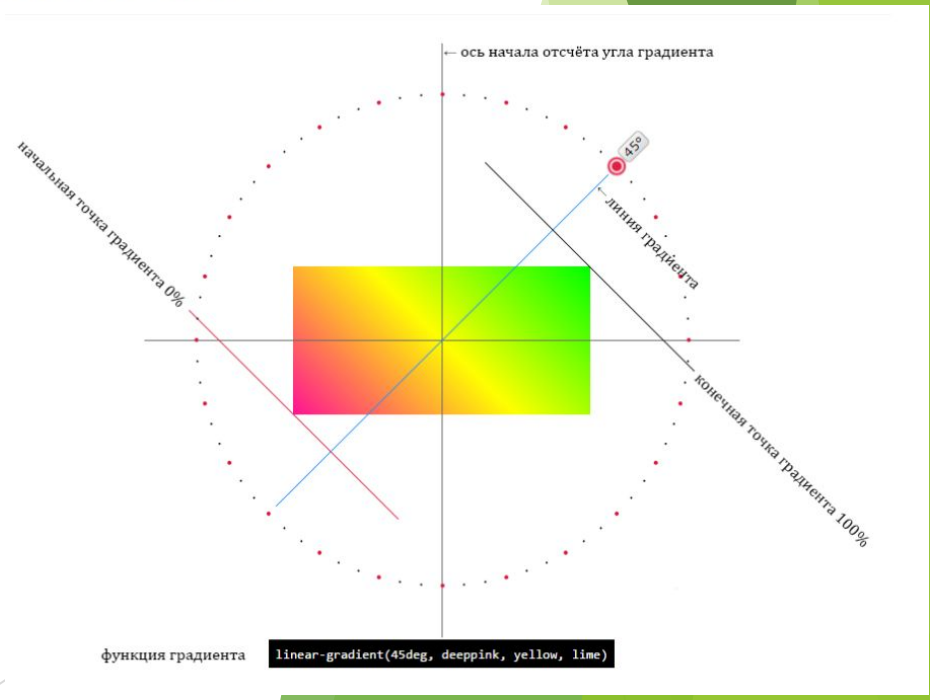
Преподаватель кафедры инженерии программного обеспечения ЖГТУ

Линейный градиент

Линейный градиент создается с помощью двух и более цветов, для которых задано направление, или **линия градиента.**

Если направление не указано, используется значение по умолчанию — **сверху-вниз.**

Цвета градиента по умолчанию распределяются равномерно в направлении, перпендикулярном линии градиента.



Линейный градиент

background: linear-gradient(*угол / сторона или угол наклона с помощью ключевого слова (пары ключевых слов), первый цвет, второй цвет и т.д.);*

Позиция	Угол	Описание	Вид
top	270deg	Сверху вниз.	
left	0deg	Слева направо.	
bottom	90deg	Снизу вверх.	
right	180deg	Справа налево.	
top left	-45deg	От левого верхнего угла к правому нижнему.	
top right	225deg	От правого верхнего угла к левому нижнему.	
bottom left	45deg	От левого нижнего угла к правому верхнему.	
bottom right	-225deg	От правого нижнего угла к левому верхнему.	

```css
div {
height: 200px;
background: linear-gradient(45deg, #EECFBA, #C5DDE8);
}
```


с помощью ключевых слов `to top`, `to right`, `to bottom`, `to left`, **которые соответствуют углу градиента, равному** `0deg`, `90deg`, `180deg` **и** `270deg` **соответственно.**

Направления градиента задаются с помощью ключевых слов: `top`, `bottom`, `left`, `right`.

Направление градиента располагается перед списком цветов и включает в себя частицу `to`. Она была добавлена в синтаксис для улучшения читабельности и наглядности:

```
background-image: linear-gradient(to right, yellow, green);
```

И сразу понятно, что это: *«Жёлто-зелёный градиент слева направо»*.

Вот примеры разных направлений градиента с цветами `yellow, green`:

`to right`	
`to left`	
`to bottom`	
`to top`	

Повторяющийся линейный градиент

Помимо обычных градиентов существуют и повторяющиеся. Их синтаксис полностью совпадает с синтаксисом обычных, только вместо `linear-gradient` пишется `repeating-linear-gradient`. Повторяющийся градиент хорош для создания полосатых фонов или фонов-орнаментов средствами CSS.

Есть несколько тонкостей, которые нужно знать про повторяющиеся градиенты:

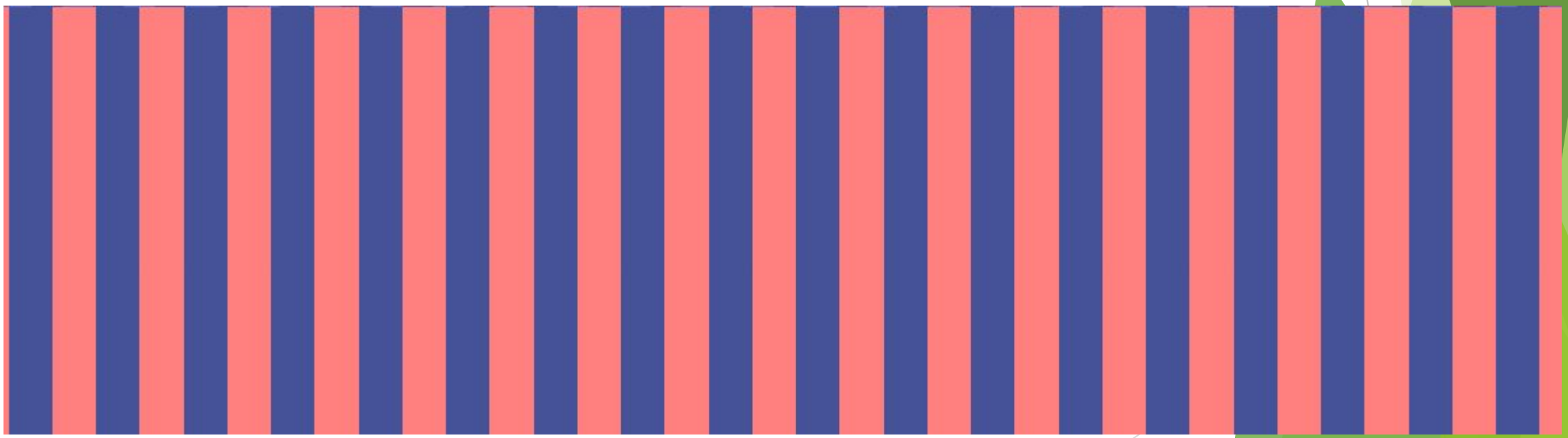
1. Размер фрагмента определяется по последнему колорстопу. Чтобы повторение было видно, последний колорстоп должен быть меньше, чем размер элемента с градиентом.
2. Если первый и последний цвета градиента различаются, то будут видны резкие границы между повторяющимися фрагментами. Чтобы от них избавиться, нужно задавать одинаковый первый и последний цвета.
3. Колорстопы в повторяющихся градиентах обычно задают в пикселях, но можно использовать и проценты.

Кстати, вместо повторяющихся градиентов можно использовать обычные градиенты в сочетании со свойствами `background-size` и `background-repeat`. Но повторяющиеся градиенты удобнее и требуют меньше кода.

В этом задании мы будем делать из обычных градиентов повторяющиеся. Обратите внимание на то, какого размера получаются фрагменты. Например, в первых двух блоках поместится ровно четыре фрагмента.

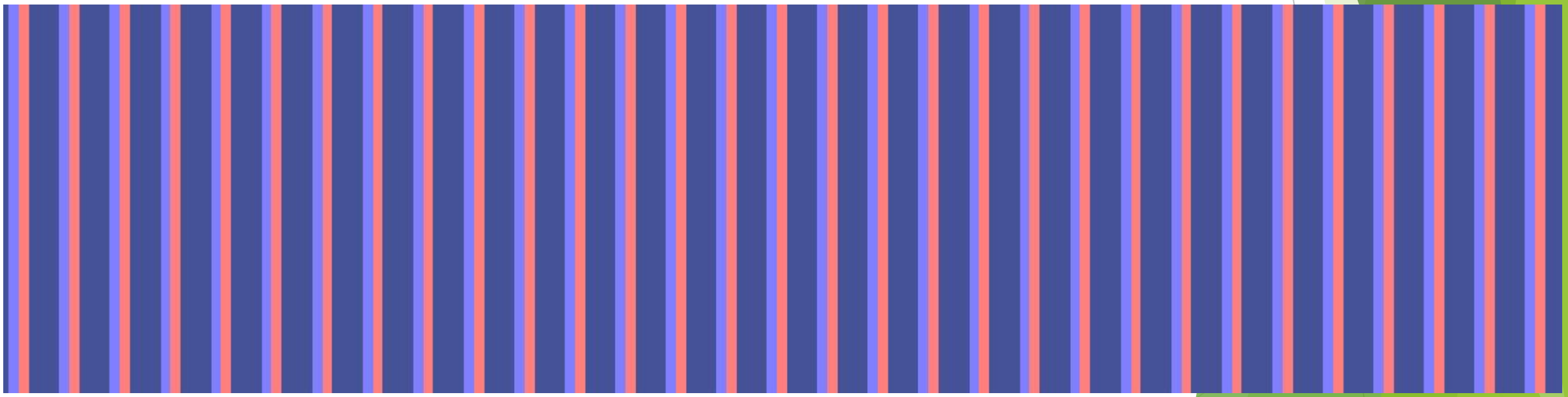
```
div {
    height: 200px;
    background: repeating-linear-gradient(45deg,
#606dbc, #606dbc 10px, #465298 10px, #465298 20px);
}
```

```
.three {
    height: 200px;
    background:
repeating-linear-gradient(90deg,
rgba(255,0,0,0.5),rgba(255,0,0,0.5) 5px,
#465298 5px, #465298 20px,
    rgba(0,0,255,0.5) 20px,rgba(0,0,255,0.5)
25px);
}
```



```
.fout {
    height: 200px;
    background:
repeating-radial-gradient(circle,
#B9ECFE, #B9ECFE 10px, #82DDFF 10px,
#82DDFF 20px);
}
```

Задание



Пропорции цветов и колорстопы

По умолчанию цвета в градиентах распределяются равномерно, в одинаковых пропорциях, но этим поведением можно управлять.

Делается это с помощью так называемых *колорстопов*, которые записываются сразу после значений цветов, например, `red 0%, yellow 100%`.

Колорстоп указывает положение цвета в градиенте, его можно задавать в процентах, пикселях и других единицах. Давайте рассмотрим несколько примеров, чтобы понять поведение колорстопов:

`yellow, green`	
`yellow 0%, green 100%`	
`yellow 0%, green 50%`	
`yellow 50%, green 100%`	
`yellow 25%, green 75%`	
`red, yellow, green`	
`red 0%, yellow 50%, green 100%`	
`red 33%, yellow 66%, green 100%`	

Позиция цвета (или колорстоп) задаёт расположение центральной части цвета, ту точку, от которой начинается переход в другой цвет.

А что будет, если задать для соседних цветов одну и ту же позицию? В этом случае получится резкий переход цветов, так как они оба будут «вытекать» из одной точки в противоположных направлениях.

Легче продемонстрировать это поведение на примере:

`yellow 25%, green 75%`	
`yellow 45%, green 55%`	
`yellow 49%, green 51%`	
`yellow 50%, green 50%`	

Этот приём часто используют для создания интересных эффектов.

Задание

Радиальный градиент `radial-gradient()`

Радиальный градиент отличается от линейного тем, что цвета выходят из одной точки (центра градиента) и равномерно распределяются наружу, рисуя форму круга или эллипса.

```
background: radial-gradient(форма градиента / размер / позиция центра, первый цвет, второй цвет и т.д.);
```

```
div {
height: 200px;
background: radial-gradient(white, #FFA9A1);
}
```

Позиция центра задаётся с помощью ключевых слов, используемых в свойстве `background-position`, с добавлением приставки `at`. Если позиция центра не задана, используется значение по умолчанию `at center`.

CSS

```css
div {
height: 200px;
background: radial-gradient(at top, #FEFFFF, #A7CECC);
}
```

С помощью пары значений, указанных в единицах длины `%`, `em` или `px`, можно управлять размером эллипсообразного градиента. Первое значение задает ширину эллипса, второе — высоту.

CSS

```
div {
height: 200px;
background: radial-gradient(40% 50%, #FAECD5, #CAE4D8);
}
```

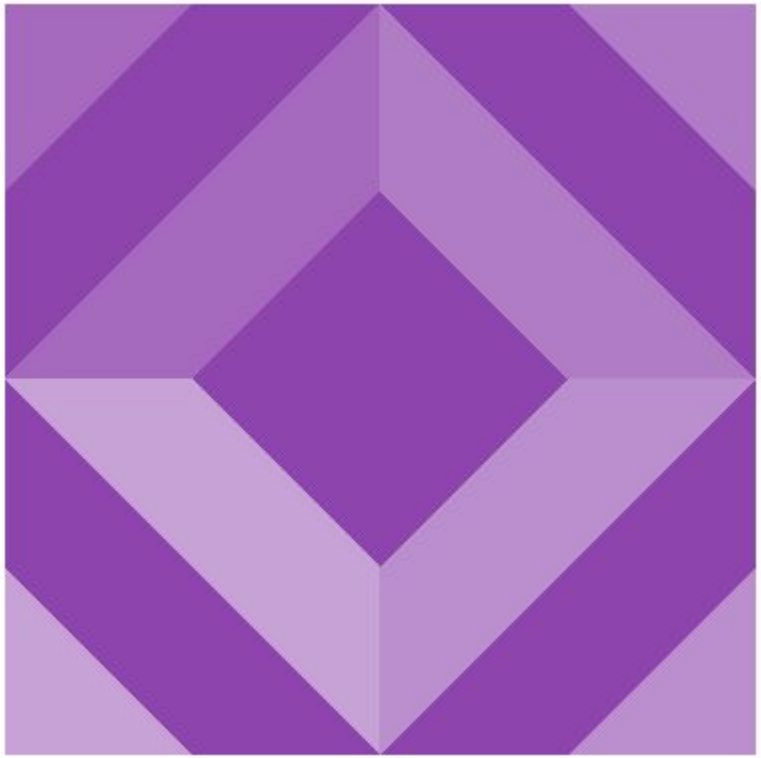
Размер градиента задаётся с помощью ключевых слов. Значение по умолчанию `farthest-corner` (к дальнему углу).

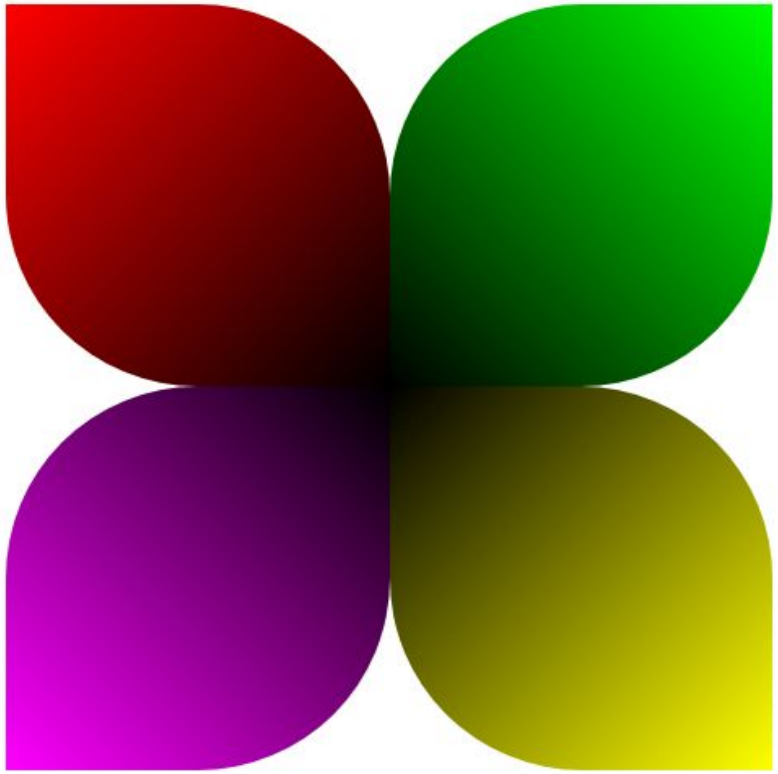
Значение	Описание
`closest-side`	Размер градиента рассчитывается из расстояния до любой ближней стороны блока для `circle` или до ближних сторон по `x` и по `y` для `ellipse`.
`farthest-side`	Размер рассчитывается из расстояния до дальних сторон.
`closest-corner`	Размер рассчитывается из расстояния до ближних углов.
`farthest-corner`	Размер рассчитывается из расстояния до дальних углов.

CSS

```
div {
height: 200px;
background: radial-gradient(circle farthest-corner at 100px 50px, #FBF2EB, #352A3B);
}
```

Задание





Строим сложный фон

Вернёмся к множественным фонам. Вы можете не только задавать несколько фоновых изображений, но и управлять размерами, расположением и повторением каждого из них по отдельности. Такая запись:

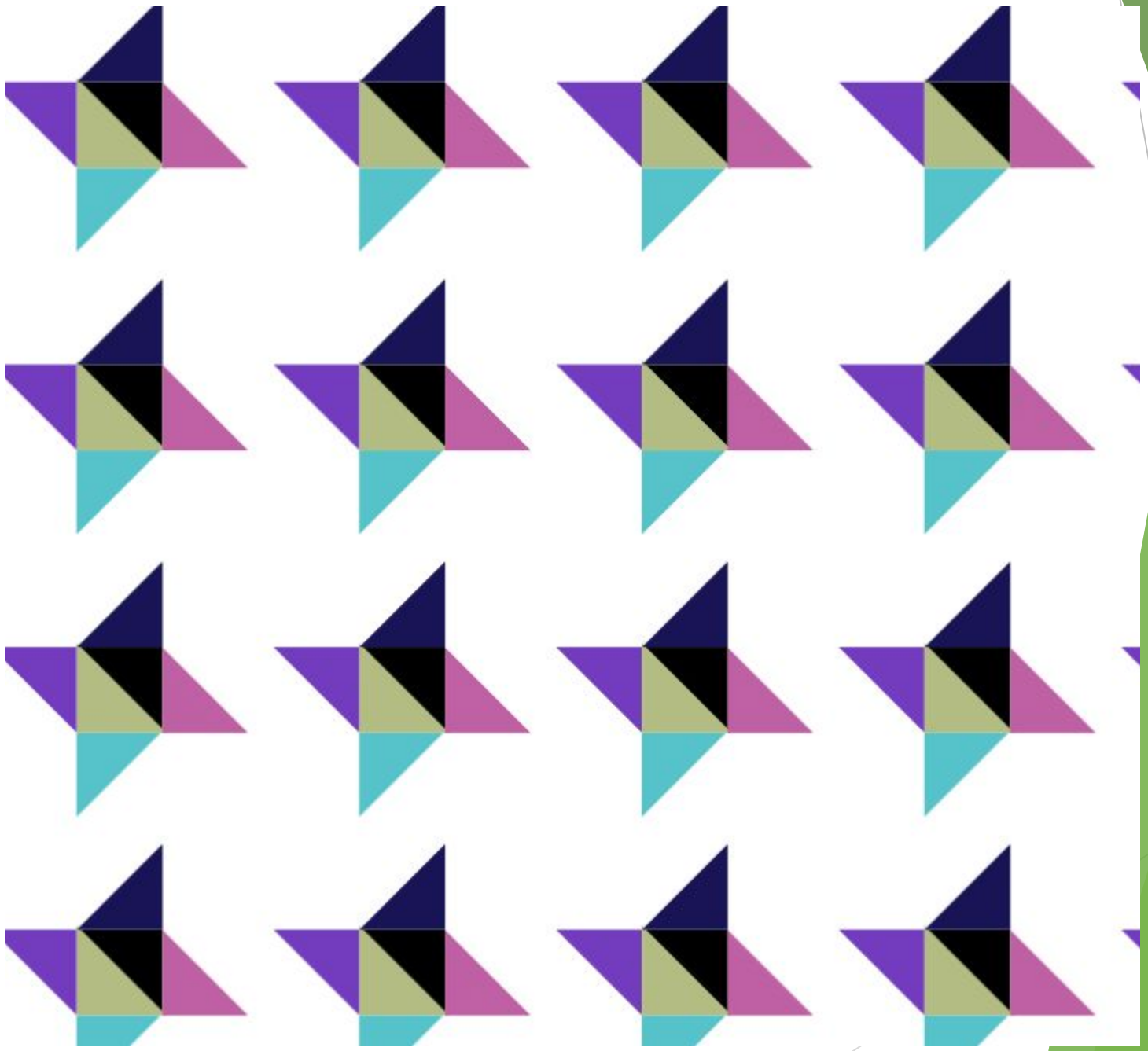
```
background-size: 100px 200px;
```

Задаст всем фоновым изображениям один и тот же размер. А такая запись:

```
background-size:
    100px 200px,
    200px 300px,
    300px 400px;
```

Задаст всем фоновым изображениям разные размеры.

Чтобы задавать разные значения свойств для множественных фонов, вам нужно просто перечислять эти значения через запятую в том же порядке, в каком вы записали изображения.



Фон

Два интересных, но малоизвестных значения привычного свойства `background-repeat`, которое задаёт повторение фона, — это `round` и `space`.

Значение свойства по умолчанию `background-repeat: repeat` просто повторяет фоновую картинку по всей ширине и высоте блока. Если части повторяющейся картинки не помещаются в ширину блока, то они просто обрезаются.

Если задать значение `background-repeat: round`, то повторяющиеся картинки по краям блока обрезаться не будут, а равномерно растянутся или сожмутся по всей ширине, чтобы занять оставшееся пространство.

Кстати, `background-repeat` принимает в качестве значения два аргумента: режим повторения по горизонтали и по вертикали. Если передать один параметр, то он применится к обоим направлениям. Например:

```
/* повторение фона repeat по горизонтали и вертикали */
background-repeat: repeat;

/* повторение фона round по горизонтали и repeat по вертикали */
background-repeat: round repeat;
```

Давайте опробуем режим повторения `round` на практике.

Позиция фона от разных сторон

Интересная возможность `background-position`, о которой мы раньше не рассказывали — расположение фона можно задавать относительно любого угла блока, а не только от левого верхнего.

Чтобы указать от какой стороны отсчитывать расположение фона, нужно перед значением координат задать ключевые слова: `top`, `right`, `bottom` или `left`. Например:

```
/* по умолчанию координаты задаются для левого верхнего угла */
background-position: 10px 50px; /* слева 10px, сверху 50px */

background-position: right 30px bottom 60px; /* справа 30px, снизу 60px */
background-position: left 50px bottom 10px; /* слева 50px, снизу 10px */
background-position: right 40px top 30px; /* справа 40px, сверху 30px */
```

Поддержка данных значений свойства `background-position` в современных браузерах практически полная.

Ещё два значения свойства `background-size` — это `contain` и `cover`.

Значение `contain` работает так:

- пропорции изображения сохраняются;
- изображению задаются максимально возможные размеры, при которых оно и по ширине, и по высоте полностью помещается в границы фона;
- изображение может не закрывать всю фоновую область блока, если пропорции изображения и блока разные.

Значение `cover` работает иначе:

- пропорции изображения сохраняются;
- изображению задаются минимально возможные размеры, при которых оно закроет всю фоновую область блока;
- если пропорции изображения и блока разные, то часть изображения обрезается.

Изображение рамки: border-image-source

Итак, семейство свойств `border-image` задаёт фоновое изображение для рамки блока. Поддержка данного семейства свойств в современных браузерах довольно неплохая.

Свойство `border-image-source` задаёт путь к изображению рамки. По умолчанию картинкой заполнятся только углы рамки. В следующих заданиях мы разберём, как можно управлять отображением рамки.

В качестве изображения для рамки используем вот такую картинку:

Синтаксис свойства такой же, как у `background-image`, то есть:

```
border-image-source: url("image.jpg");
```

Давайте же зададим фоновое изображение рамки и начнём его настраивать.

Изображение рамки: border-image-repeat

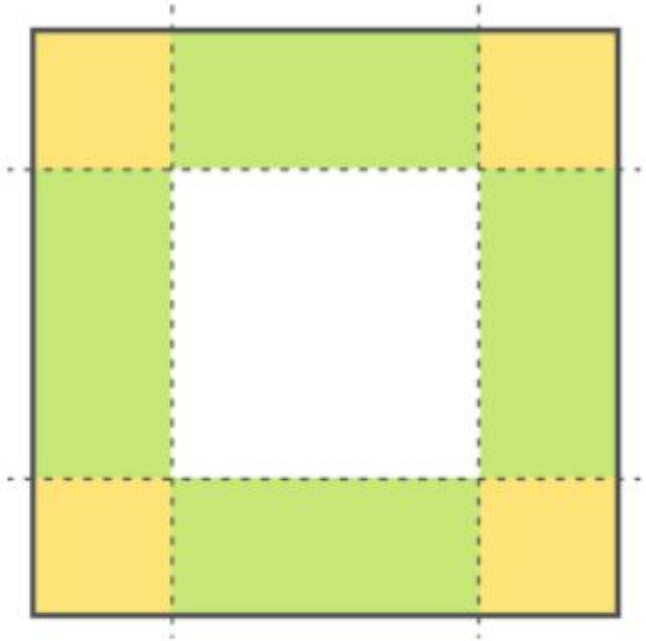
Свойство `border-image-repeat` задаёт способ заполнения фоном боковых сторон рамки (зелёные области на рисунке).

У свойства четыре значения: `stretch`, `repeat`, `space` и `round`.

Значение по умолчанию — `stretch`. При этом значении фоновые картинки растягиваются на всю длину боковых сторон.

Если задано значение `repeat`, то фоновые картинки будут повторяться. При этом они могут обрезаться.

Можно устанавливать режим заполнения отдельно для горизонтальных и вертикальных сторон рамки.



Например:

```
/* все стороны рамки заполняются в режиме stretch */
border-image-repeat: stretch;

/* горизонтальные стороны — режим repeat, вертикальные — stretch */
border-image-repeat: repeat stretch;
```

Задание

- Вставить свой портрет в рамку

Изображение рамки: border-image-outset

Ещё одно свойство, относящееся к фоновому изображению рамки, `border-image-outset`. Аналогично `outline-offset` это свойство позволяет отодвинуть рамку за пределы элемента, но при этом одновременно немного масштабируя картинку. Отрицательные значения `border-image-outset` не поддерживаются.

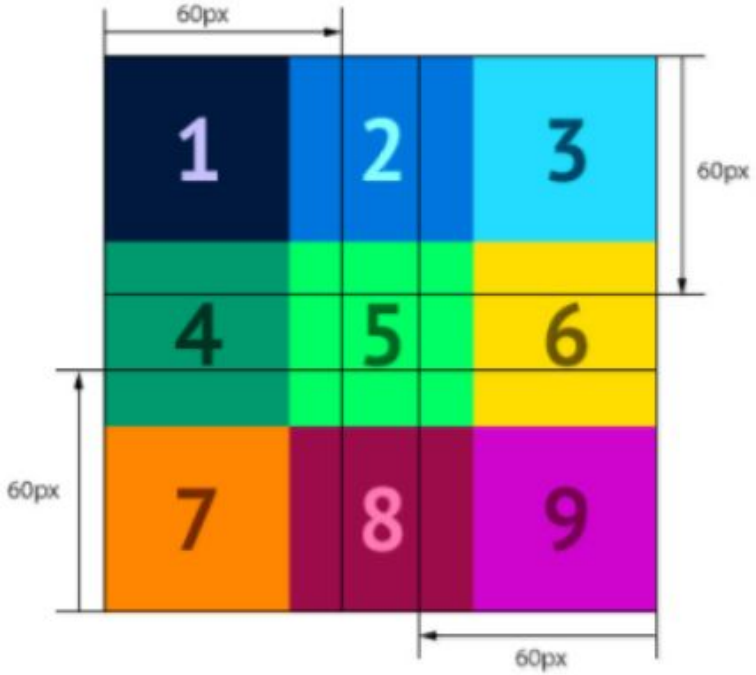
Отступы рамок-изображений тоже можно задавать разные для каждой из сторон. Синтаксис обычный:

```
border-image-outset: 10px;
border-image-outset: 10px 20px 30px 40px;
```

Изображение рамки: border-image-slice

Давайте разберёмся, как работает механизм «нарезки» фонового изображения для рамки.

Каждая рамка имеет `9` областей: `4` угла, `4` стороны и центральную область. Для заполнения этих областей браузер должен нарезать картинку для рамки на `9` частей. Когда браузер не знает, как это сделать, он просто размещает картинку по углам — мы видели это в предыдущем задании.



Свойство `border-image-slice` задаёт отступы от краёв картинки до четырёх линий, которые «разрезают» её на части, как на схеме справа. Если эти отступы небольшие, то получается «нарезка» из `9` частей, которые затем размещаются в соответствующих областях рамки.

Но если отступы слишком большие (больше половины картинки), то браузер не может получить `9` частей и располагает то, что отрезалось по углам.

Значение свойства можно задавать числом без единицы измерения (оно обычно обозначает пиксели) или в процентах (относительно размера самой картинки). Пример:

```
border-image-slice: 60;
border-image-slice: 10%;
```

Нарезка несимметричных картинок

Сравните две картинки:

Для нарезки первой из них можно было задать одинаковые отступы линий разреза — `50px`. Для второй картинки этого явно недостаточно — она менее симметрична.

С помощью `border-image-slice` можно задавать разные отступы линий разреза. Для этого нужно задавать значения через пробел в порядке: верх, право, низ, лево. Пример:

```
border-image-slice: 10 20 30 40;
```

Средняя часть картинки обычно не используется. Но если в значение свойства добавить ключевое слово `fill`, то средняя часть картинки будет отображаться в средней области рамки: она закроет собой фон блока, но не закроет содержимое. Пример:

```
border-image-slice: 10 20 30 40 fill;
```

Изображение рамки: border-image-width

Следующее свойство, которое мы рассмотрим — `border-image-width`.

У блока должна существовать рамка определённой толщины `border-width`, тогда ему можно задать и фоновую картинку для рамки. Область, в которой будет отображаться эта картинка по умолчанию равна ширине рамки.

Свойство `border-image-width` позволяет управлять шириной видимой области рамки-картинки, масштабировать её. Саму ширину рамки это свойство не меняет.

Если значение этого свойства больше `border-width`, картинка рамки заползёт под содержимое, даже если не задано свойство `fill`.

Ширина рамки-картинки задаётся в `%`, `px`, `em` или других единицах измерения. Также возможно значение `auto`, при котором ширина зависит от значения `border-image-slice`.

Можно задавать разную ширину сторон. В этом случае значения перечисляются аналогично `margin`, `padding` в последовательности: верхнее, правое, нижнее, левое. Например:

```
border-image-width: 10px 20px 30px 40px;
border-image-width: 10px 50px;
```

Попробуем поуправлять шириной рамки-картинки.

Задание

При наведении на картинку снять маску и очки

Рисование

Используя рамки и свойства типа

```
border-right-color: #2ecc40;
```

Нарисуйте