1. Как создать объект класса String, какие конструкторы класса String вы знаете? Что такое строковый литерал? Объясните, что значит “упрощенное создание объекта String”?

Создать объект класса String можно при помощи оператора new.

```
String(byte[] bytes)
String(char[] value)
String(String original)
String(StringBuilder builder)
String(StringBuffer buffer)
```

Любой текст в Java является строковым литералом. Упрощённое создание String не требует от нас явного использования оператора new:

String text = “Some text”;

2. Можно ли изменить состояние объекта типа String? Что происходит при попытке изменения состояния объекта типа String? Можно ли наследоваться от класса String? Как вы думаете, почему строковые объекты immutable?

Нет. Объект типа String неизменяемый. При попытке изменения состояния мы получаем уже новый объект с новым значением. От класса String нельзя наследоваться, т.к. он объявлен как final.

1) Строки immutable т.к. они широко используются в Java. Например, при работе с сетью или открытии файлов. Если бы строки были изменяемыми, то злоумышленник мог бы поменять её значение во время передачи и получить доступ к другому файлу или серверу.
2) Из-за того, что строка не изменяется, она кэширует свой хэшкод и не вычисляет его каждый раз, когда мы его вызываем, что делает строку очень быстрой как ключ для hashmap.
3) immutable делает экземпляры строк thread-safe.

3. Объясните, что такое кодировка? Какие кодировки вы знаете? Как создать строки в различной кодировке?

Кодировка — это таблица, в которой описывается соответствие определённого символа и числа. Utf-8, CP1251. У String есть конструктор, который в качестве параметра принимает нужную нам кодировку и преобразует текст в эту кодировку.

4. Что такое пул литералов? Как строки заносятся в пул литералов? Как занести строку в пул литералов и как получить ссылку на строку, хранящуюся в пуле литералов? Где хранится (в каком типе памяти) пул литералов в Java 1.6 и Java 1.7?

Виртуальная машина Java хранит строки созданные при помощи двойных кавычек в пуле строк. И если мы создаём новую строку и в кавычках указываем текст, то первоочерёдно JVM ищет эту строку в пуле строк. Если она там есть – то возвращает ссылку на эту строку, иначе – создаётся новый объект с полученным значением и сохраняется в пул.

Когда мы используем оператор new, виртуальная машина создает объект String, но не хранит его в пуле строк. Мы можем использовать метод intern() для сохранения строки в пуле

строк, или получения ссылки, если такая строка уже находится в пуле.

До JDK 7 пул строк хранился в отдельном пуле (PermGen), который имел фиксированный размер, и сборщик мусора его не обрабатывал. В результате был риск возникновения ошибки outOfMemory, если мы добавим слишком много стрингов.

С версии JDK 7 пул строк хранится в общей куче. Теперь сборщик мусора может удалять ненужные строки.

В JDK 9 String уже не обязательно будет создан из char[] он может при необходимости быть создан из byte[].

5. В чем отличие объектов классов StringBuilder и StringBuffer от объектов класса String? Какой из этих классов потокобезопасный? Как необходимо сравнивать на равенство объекты классов StringBuilder и StringBuffer и почему?

StringBuilder и StringBuffer изменяемые в отличие от String. StringBuffer ещё и потокобезопасен.

StringBuilder наследует equals() от Object. А значит этот метод вернёт true только в случае, если мы сравниваем один и тот же объект. Поэтому для сравнения двух объектов StringBuilder можно сначала вызвать метод toString() а затем уже .equals()

6. Что такое Unicode?

Это самый распространённый в мире стандарт кодирования различных символов.

7. Какие методы класса String используются для работы с кодовыми точками? Как вы думаете, когда следует их использовать?

Методы `codePointAt` `codePointCount` и другие. Их следует использовать, если метод length работает не верно.

Regular Expressions

1. **Расскажите, что представляет собой регулярное выражение? Что такое метасимволы регулярного выражения? Какие вы знаете классы символов регулярных выражений? Что такое квантификаторы? Какие логические операторы регулярных выражений вы знаете? Что значит “якорь” для регулярного выражения?**

Регулярное выражение – это шаблон поиска некоторой строки в тексте.

Для написания шаблона регулярного выражения используются как обычные цифровые и буквенные символы, так и метасимволы – символы, имеющие специальное назначение.

Квантификатор – это метасимвол, определяющий сколько раз искать шаблон.

Классы символов:

- Простые классы символов [abc]
- Инвертированные классы символов [^abc]
- Диапазонные классы символов [a-c]

Логические операторы AND, OR, NOT.

AND – встроен, например в выражение John – означает и букву J и букву o и т.д.

OR – обозначается символом |

NOT – символов ^

Якорями называются метасимволы начало строки (^) и конца ($)

2. **Какие java-классы работают с регулярными выражениями? В каком пакете они расположены? Приведите пример анализа текста с помощью регулярного выражения и поясните код примера.**

API регулярных выражений состоит из трех классов: Pattern, Matcher и PatternSyntaxException, расположенных в пакете java.util.regex

```
String text = "Егор Алла Александр";
    Pattern pattern = Pattern.compile("A.+a");
    Matcher matcher = pattern.matcher(text);
    while (matcher.find()) {
        System.out.println(text.substring(matcher.start(),
matcher.end()));
    }
```

3. **Что такое группы в регулярных выражениях? Как нумеруются группы? Что представляет собой группа номер 0(ноль)? Приведите пример с использованием групп регулярного выражения.**

Группы в регулярных выражениях позволяют сохранить найденный набор символов для дальнейшего использования. Группы нумеруются слева направо начиная с 1. Группа 0 представляет собой выражение в полном виде.

```
String text = "Егор Алла Александр";
    Pattern pattern = Pattern.compile("A.+a");
    Matcher matcher = pattern.matcher(text);
    if (matcher.find()) {
        System.out.println("Первое соответствие шаблону " +
matcher.group(1));
    } else {
       System.out.println("Ничего не найдено");
}
```