1.Назовите классы, которые используются для создания меню. Перечислите свойства и методы.

Для создания меню используются классы *MainMenu, ContextMenu*и *MenuItem,*порожденные от абстрактного класса Menu. В классе Menu объявлен внутренний класс MenuItemCollection, который наследуется классами MainMenu, ContextMenu и MenuItem.

Наиболее важные свойства компонента MenuStrip:

* Dock: прикрепляет меню к одной из сторон формы
* LayoutStyle: задает ориентацию панели меню на форме. Может также, как и с ToolStrip, принимать следующие значения
  + HorizontalStackWithOverflow: расположение по горизонтали с переполнением - если длина меню превышает длину контейнера, то новые элементы, выходящие за границы контейнера, не отображаются, то есть панель переполняется элементами
  + StackWithOverflow: элементы располагаются автоматически с переполнением
  + VerticalStackWithOverflow: элементы располагаются вертикально с переполнением
  + Flow: элементы размещаются автоматически, но без переполнения - если длина панели меню меньше длины контейнера, то выходящие за границы элементы переносятся
  + Table: элементы позиционируются в виде таблицы
* ShowItemToolTips: указывает, будут ли отображаться всплывающие подсказки для отдельных элементов меню
* Stretch: позволяет растянуть панель по всей длине контейнера
* TextDirection: задает направление текста в пунктах меню

2. Что может содержать строка состояния? Какие есть методы управления строкой состояния?

**Строка** **состояния** — это специальная область внутри первичного окна (обычно в нижней его части), предназначенная для вывода информации о текущем состоя­ рументов, **строка** **состояния** **может** **содержать** элементы **управления**; тем не менее, рекомендуется включать в нее информацию «только для чтения» или не интерак­.

3. Что такое регулярные выражения? Где и как их можно использовать?

**Регулярные** **выражения** - это шаблоны, **используемые** для сопоставления последовательностей символов в строках.

**Регулярное** **выражение** **можно** **использовать** для определения повторяющихся слов,

4. Что такое привязки (якоря) в RegEx? Приведите примеры.

Привязки (или атомарные утверждения нулевой ширины) указывают положение в строке, где должно быть найдено соответствие. У символов каретки ^ и доллара $ есть специальные значения в регулярных выражениях. Они называются «якоря» (anchors). Каретка ^ означает совпадение с началом текста, а доллар $ – с концом.

5. Зачем используют конструкции группирования? Приведите примеры.

Конструкции группирования отображают части выражений регулярных выражений и захватывают части строки входной строки. Конструкции группирования можно использовать в следующих целях.

* Сопоставление с частью выражения, которая повторяется во входной строке.
* Применение квантификатору к части выражения с несколькими языковыми элементами регулярных выражений. Дополнительные сведения о квантификаторах см. в разделе [Quantifiers](https://docs.microsoft.com/ru-ru/dotnet/standard/base-types/quantifiers-in-regular-expressions).
* Добавление в строку части выражения, которая возвращается методами [Regex.Replace](https://docs.microsoft.com/ru-ru/dotnet/api/system.text.regularexpressions.regex.replace) и [Match.Result](https://docs.microsoft.com/ru-ru/dotnet/api/system.text.regularexpressions.match.result) .
* Извлечение отдельных частей выражения из свойства [Match.Groups](https://docs.microsoft.com/ru-ru/dotnet/api/system.text.regularexpressions.match.groups) и обработка их по отдельности от всего текста.

6. Что такое квантор или множители? Приведите примеры.

**Квантор** – логическая операция, ограничивающая область истинности предиката и создающая высказывание

Квантификаторы определяют количество экземпляров символа, группы или класса символов, которое должно присутствовать во входных данных, чтобы было зафиксировано совпадение. В приведенной ниже таблице перечислены квантификаторы, поддерживаемые платформой .NET.

| **КВАНТОРЫ В РЕГУЛЯРНЫХ ВЫРАЖЕНИЯХ** | | |
| --- | --- | --- |
| **Жадный квантификатор** | **Ленивый квантификатор** | **Описание** |
| \* | \*? | Совпадение ноль или несколько раз. |
| + | +? | Совпадение один или несколько раз. |
| ? | ?? | Совпадение ноль или один раз. |
| { n } | { n }? | Совпадение ровно n раз. |
| { n ,} | { n ,}? | Совпадение как минимум n раз. |
| { n , m } | { n , m }? | Совпадение от n до m раз. |

Количества n и m являются целочисленными константами. Обычно квантификаторы "жадные" — они заставляют обработчик регулярных выражений выделить максимально возможное число вхождений определенного шаблона. Добавление символа ? в квантификатор делает его отложенным (или ленивым). Это приводит к тому, что обработчик регулярных выражений пытается сопоставить так мало вхождений, как это возможно. Полное описание различий между "жадными" и "ленивыми" квантификаторами см. в разделе [Жадные и ленивые квантификаторы](https://docs.microsoft.com/ru-ru/dotnet/standard/base-types/quantifiers-in-regular-expressions#Greedy) далее в этой статье.

7. Напишите регулярное выражение для проверки номера MTC (Velcom).

/(?:\+|\d)[\d\-\(\) ]{9,}\d/g

8. Напишите регулярное выражение для проверки паспорта.

^(?!^0+$)[a-zA-Z0-9]{3,20}$

9. Напишите регулярное выражение для проверки даты.

 **Дата в формате YYYY-MM-DD:** [0-9]{4}-(0[1-9]|1[012])-(0[1-9]|1[0-9]|2[0-9]|3[01]) **UPD.** Более строгая проверка: (19|20)\d\d-((0[1-9]|1[012])-(0[1-9]|[12]\d)|(0[13-9]|1[012])-30|(0[13578]|1[02])-31)

 **Дата в формате DD/MM/YYYY:** (0[1-9]|[12][0-9]|3[01])[- /.](0[1-9]|1[012])[- /.](19|20)\d\d

 **Дата в формате DD.MM.YYYY:** (0?[1-9]|[12][0-9]|3[01]).(0?[1-9]|1[012]).((19|20)\d\d)

10. Напишите регулярное выражение для проверки УДК

11. Напишите регулярное выражение для проверки Фамилии.

 **Фамилия+Имя или же ФИО полностью**

 ^[а-яА-ЯёЁa-zA-Z]+ [а-яА-ЯёЁa-zA-Z]+ ?[а-яА-ЯёЁa-zA-Z]+$

12. Напишите регулярное выражение для проверки пароля, логина.

 **Пароль** (Строчные и прописные латинские буквы, цифры): ^(?=.\*\d)(?=.\*[a-z])(?=.\*[A-Z])(?!.\*\s).\*$

 **Пароль** (Строчные и прописные латинские буквы, цифры, спецсимволы. Минимум 8 символов): (?=^.{8,}$)((?=.\*\d)|(?=.\*\W+))(?![.\n])(?=.\*[A-Z])(?=.\*[a-z]).\*$

13. Разберите ?<=**Ошибка! Не указано имя файла.**)" "[a-zA-Zа-яА-Я,-;:]{5,50}" /^[a-z0-9\_-]{3,16}$/ /^#?([a-f0-9]{6}|[a-f0-9]{3})$/ /^(https?:\/\/)?([\da-z\.-]+)\.([a-z\.]{2,6})([\/\w \.-]\*)\*\/?$/ /^(?:(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.){3}(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0- 9]?)$/ /^<([a-z]+)([^(.\*)|\s+\/>)$/