1.

ООП – это метод программирования, состоящий из совокупности объектов, каждый из которых является экземпляром определённого класса

Абстра́кция  — в объектно-ориентированном программировании это придание объекту характеристик, которые отличают его от всех других объектов, четко определяя его границы.

Инкапсуляция - это свойство объекта/класса регулировать доступ к определенным своим компонентам извне самого объекта/класса. Например, на мониторе есть кнопки регулировки яркости - они в прямом доступе для пользователя. А есть микросхемы внутри - к ним доступ ограничен. Это и есть пример инкапсуляции. Объект монитор регулирует доступ к своим компонентам.  
  
Наследование - это свойство классов расширяться (Это создание нового объекта на основе существющего). Например, у нас есть класс животное. У животных есть глаза и нос. Класс собака - унаследован от объекта животное. Это значит, что у собаки также есть глаза и нос, но в добавок она еще и может гавкать, чего не может другое животное (не собака). Класс кошка, например, также унаследован от объекта животное.  
  
Полиморфизм - это следствие наследования. Это свойство унаследованных объектов иметь одинаковые методы, которые будут работать по-разному в контексте объектов. Например, у нас есть класс фигура и классы квадрат, треугольник и трапеция - унаследованные от фигуры. Каждая фигура содержит функционал для вычисления площади, но у каждой фигуры он свой.

2.

Класс в ООП – это основной элемент, в котором находится вся информация об объекте. Объект это экземпляр класса, проще говоря из одного класса можно создать бесконечное количество объектов.

3.

Перегрузка функций – это функции с одним и тем же названием но с разным набором параметров и в зависимости от имеющихся параметров применять определённую версию функции.

4.

Функция которая возвращает значение, например:

int func (int a, int b)

{

int c = a + b;

return c;

}

Функция которая не возвращает значение, например:

Void func()

{

Console.WriteLine(“Hello”);

}

5.

-  имя представителя класса.имя элемента;

-  имя класса::имя элемента;

-  указатель на представитель класса->имя элемента

private:

  /\* список свойств и методов для использования внутри класса \*/

  public:

  /\* список методов доступных другим функциям и объектам программы \*/

  protected:

  /\*список средств, доступных при наследовании\*/

6.

Наследование - это свойство классов расширяться (Это создание нового объекта на основе существющего). Например, у нас есть класс животное. У животных есть глаза и нос. Класс собака - унаследован от объекта животное. Это значит, что у собаки также есть глаза и нос, но в добавок она еще и может гавкать, чего не может другое животное (не собака). Класс кошка, например, также унаследован от объекта животное.  
7.

Бывают явное и не явное преобразование типа:

Явное:

int a = 4;

int b = 6;

byte c = (byte)(a+b);

Не явное:

byte a = 4;

ushort b = a;

8.

Обработка исключений – это механизм языков программирования, предназначенный для описании реакции программы на ошибку во время выполнения программы

9.

конструктор (от слова construct — создавать) – это специальный метод класса, который предназначен для инициализации элементов класса некоторыми начальными значениями.

деструктор (от слова destruct — разрушать) — специальный метод класса, который служит для уничтожения элементов класса.

10.

Событие - это реакция программы, которая возникает в различных точках исполняемого кода при выполнении определённых условий. Например при нажатии кнопки выводится какое то сообщение, это и есть событие.

11.

Делегаты – это указатель на функцию, 1 делегат может указывать сразу на несколько функций

12.

Console.Write – вывод

Console.WriteLine – вывод на следующую строку

Console.Read – ввод

Console.ReadLine – ввод с переносом на следующую строку

13.

Close: закрывает считываемый файл и освобождает все ресурсы

Peek: возвращает следующий доступный символ, если символов больше нет, то возвращает -1

Read: считывает и возвращает следующий символ в численном представлении. Имеет перегруженную версию: Read(char[] array, int index, int count), где array - массив, куда считываются символы, index - индекс в массиве array, начиная с которого записываются считываемые символы, и count - максимальное количество считываемых символов

ReadLine: считывает одну строку в файле

ReadToEnd: считывает весь текст из файла

Пример:

string path= @"C:\SomeDir\hta.txt";

try

{

    Console.WriteLine("\*\*\*\*\*\*считываем весь файл\*\*\*\*\*\*\*\*");

    using (StreamReader sr = new StreamReader(path))

    {

        Console.WriteLine(sr.ReadToEnd());

    }

    Console.WriteLine();

    Console.WriteLine("\*\*\*\*\*\*считываем построчно\*\*\*\*\*\*\*\*");

    using (StreamReader sr = new StreamReader(path, System.Text.Encoding.Default))

    {

        string line;

        while ((line = sr.ReadLine()) != null)

        {

            Console.WriteLine(line);

        }

    }

    Console.WriteLine();

    Console.WriteLine("\*\*\*\*\*\*считываем блоками\*\*\*\*\*\*\*\*");

    using (StreamReader sr = new StreamReader(path, System.Text.Encoding.Default))

    {

        char[] array = new char[4];

        // считываем 4 символа

        sr.Read(array, 0, 4);

        Console.WriteLine(array);

    }

}

catch (Exception e)

{

    Console.WriteLine(e.Message);

}