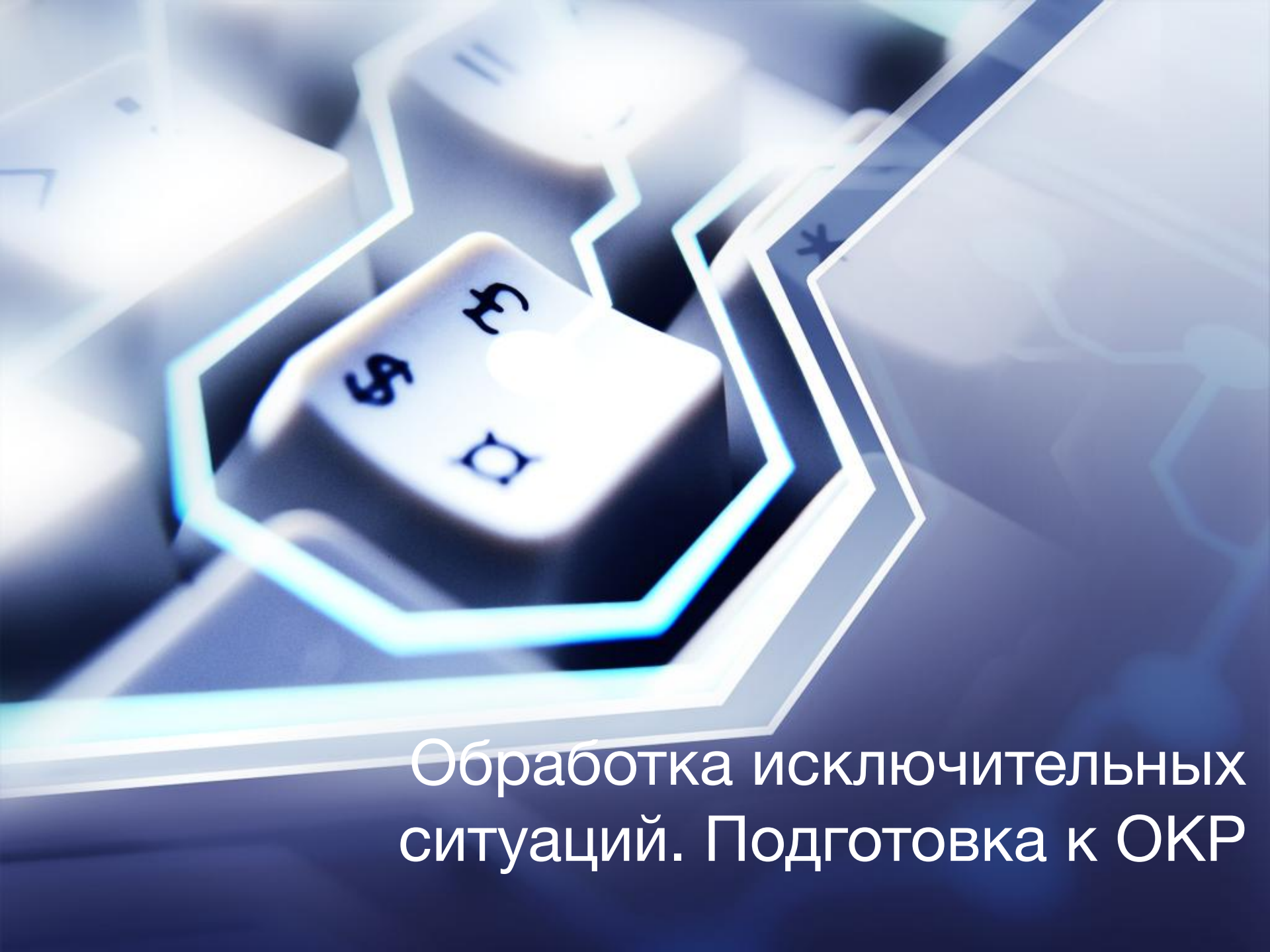

Обработка исключительных ситуаций. Подготовка к ОКР

Исключения

- В C# есть механизм, который позволяет обрабатывать подобные ошибки и таким образом избегать аварийного завершения программы. Он так и называется: *механизм обработки исключительных ситуаций (исключений).*

Родитель ошибок

- Каждому типу ошибки соответствует свое исключение. В c# исключения являются классами, которые имеют общего предка — класс Exception, определенный в пространстве имен System.

Конструкторы

Имя	Описание
Exception()	Инициализирует новый экземпляр класса Exception.
Exception(String)	Инициализирует новый экземпляр класса Exception с указанным сообщением об ошибке.

Свойства

Имя	Описание
Message	Получает сообщение, описывающее текущее исключение.
StackTrace	Получает строковое представление непосредственных кадров в стеке вызова.

Важные системные ошибки

- *ArrayTypeMismatchException* Тип сохраненного значения несовместим с типом массива
- *DivideByZeroException* Предпринята попытка деления на ноль
- *IndexOutOfRangeException* Индекс массива выходит за пределы диапазона

Важные системные ошибки

- *InvalidCastException* Некорректное преобразование в процессе выполнения
- *OutOfMemoryException* Вызов new был неудачным из-за недостатка памяти
- *Overflow/Exception* Переполнение при выполнении арифметической операции
- *StackOverflowException* Переполнение стека

Перехват(обработка исключений)

Можно задать способ обработки исключений

Стандартная обработка: вывод сообщения
И завершение программы

Обработка исключений

```
try //обязательный блок
{    Код в котором может быть исключение}
catch (SomeSpecificException ex)
{    Действия по обработке}
finally
{
    Обязательные действия
}
```

Условия для перехвата

Имеется хорошее понимание причин создания исключения, существует возможность реализовать конкретное восстановление.

Например предложить пользователю ввести новое имя файла при перехвате объекта FileNotFoundException.

Условия для перехвата

Возможность создания и вызова нового, более конкретного исключения.

```
int GetInt(int[] array, int index)
{   try
   {       return array[index];   }
   catch(System.IndexOutOfRangeException e)
   {
      throw new System.ArgumentOutOfRangeException(
         "Parameter index is out of range.");
   }}
```

Условия для перехвата

Требуется частично обработать исключение перед передачей его на дополнительную обработку.

```
try
{   // Получить доступ к ресурсу
}
catch (System.UnauthorizedAccessException e)
{   // Попытка не удалась.
    LogError(e);
     throw;
}
```

Блок catch

- Если у нас возникает исключение определенного типа, то оно переходит к соответствующему блоку catch.
- При этом более частные исключения следует помещать в начале, и только потом более общие классы исключений.

Блок catch

```
static void Main(string[] args)
{
    try
    {

    }
    catch (FileNotFoundException e)
    {
        // Обработка исключения, возникшего при отсутствии файла
    }
    catch (IOException e)
    {
        // Обработка исключений ввода-вывода
    }
    Console.ReadLine();
}
```

Генерация исключения

- Исключения могут явно генерироваться программной с помощью ключевого слова **throw.**

```
static void Main(string[] args)
{
    try
    {
        string message = Console.ReadLine();
        if (message.Length > 6)
        {
            throw new Exception("Длина строки больше 6 символов");
        }
    }
    catch (Exception e)
    {
        Console.WriteLine("Ошибка: " + e.Message);
    }
    Console.ReadLine();
}
```