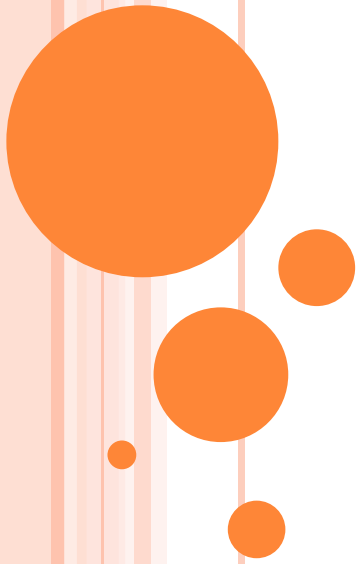


EXCEPTION HANDLING

Resources

[1] Object Oriented Programming with C++ (3rd Edition) E Balagurusamy

[2] Teach Yourself C++ (3rd Edition) H Schildt

EXCEPTIONS

- Exceptions are run time anomalies or unusual conditions that a program may encounter during execution.
- Conditions such as
  - Division by zero
  - Access to an array outside of its bounds
  - Running out of memory
  - Running out of disk space
- It was not a part of original C++.
- It is a new feature added to ANSI C++.

Exception Handling

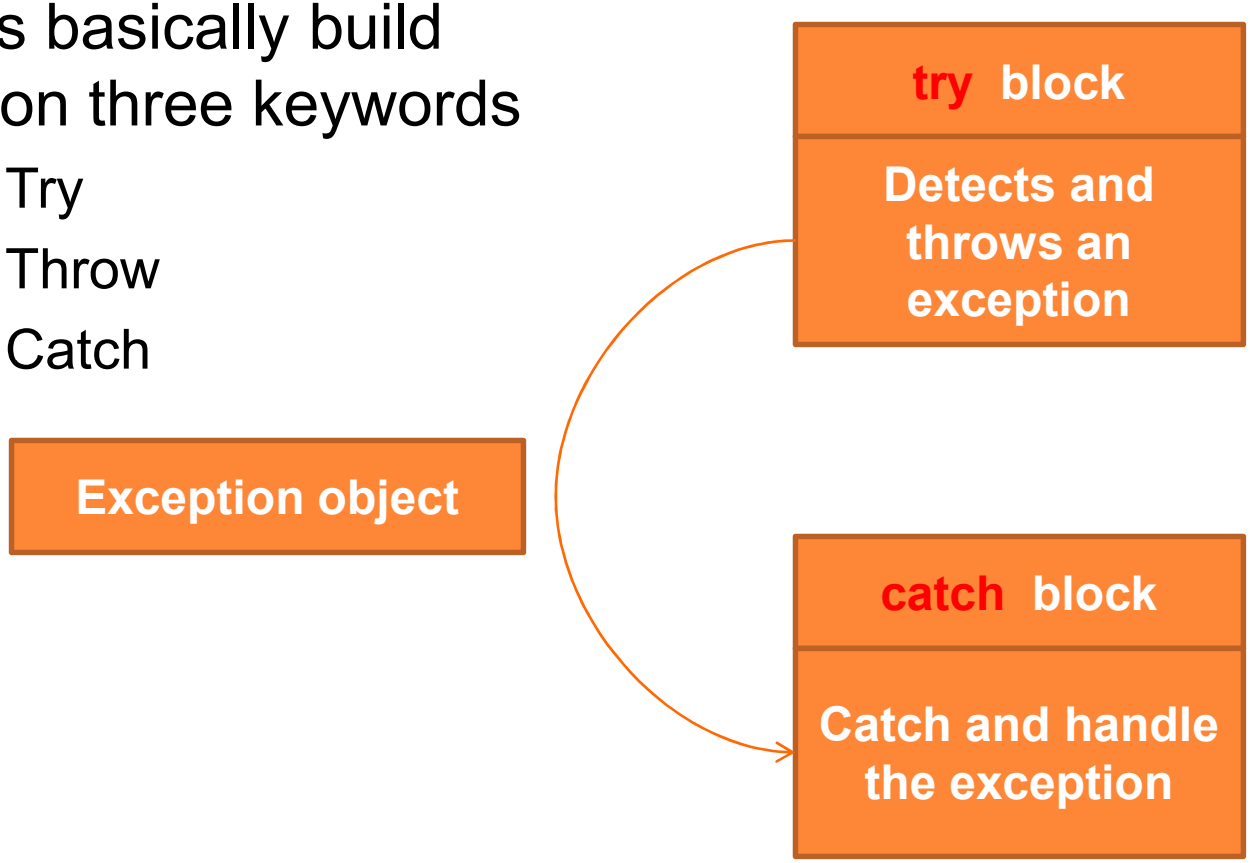
- Exceptions are of 2 kinds
  - Synchronous Exception:
    - Out of rage
    - Over flow
  - Asynchronous Exception: Error that are caused by causes beyond the control of the program
    - Keyboard interrupts
- In C++ only synchronous exception can be handled.

Exception Handling (Cont…)

- Exception handling mechanism
  - Find the problem (Hit the exception)
  - Inform that an error has occurred (Throw the exception)
  - Receive the error information (Catch the exception)
  - Take corrective action (handle the exception)

Exception Handling Mechanism

- It is basically build upon three keywords
  - Try
  - Throw
  - Catch



EXCEPTION HANDLING MECHANISM (CONT…)

- The keyword **try** is used to preface a block of statements which may generate exceptions.
- When an exception is detected, it is thrown using a **throw** statement in the try block.
- A **catch** block defined by the keyword ‘catch’ catches the exception and handles it appropriately.
- The catch block that catches an exception must immediately follow the try block that throws the exception.

EXCEPTION HANDLING MECHANISM (CONT…)

```
try
{
    …                          // Block of statements
    throw exception;           // which detect and
    …                          // throws an exception
}
catch(type arg)                // catch exception
{
    …                          // Block  of statement
    …                          // that handles the
    …                          // exception
}
```

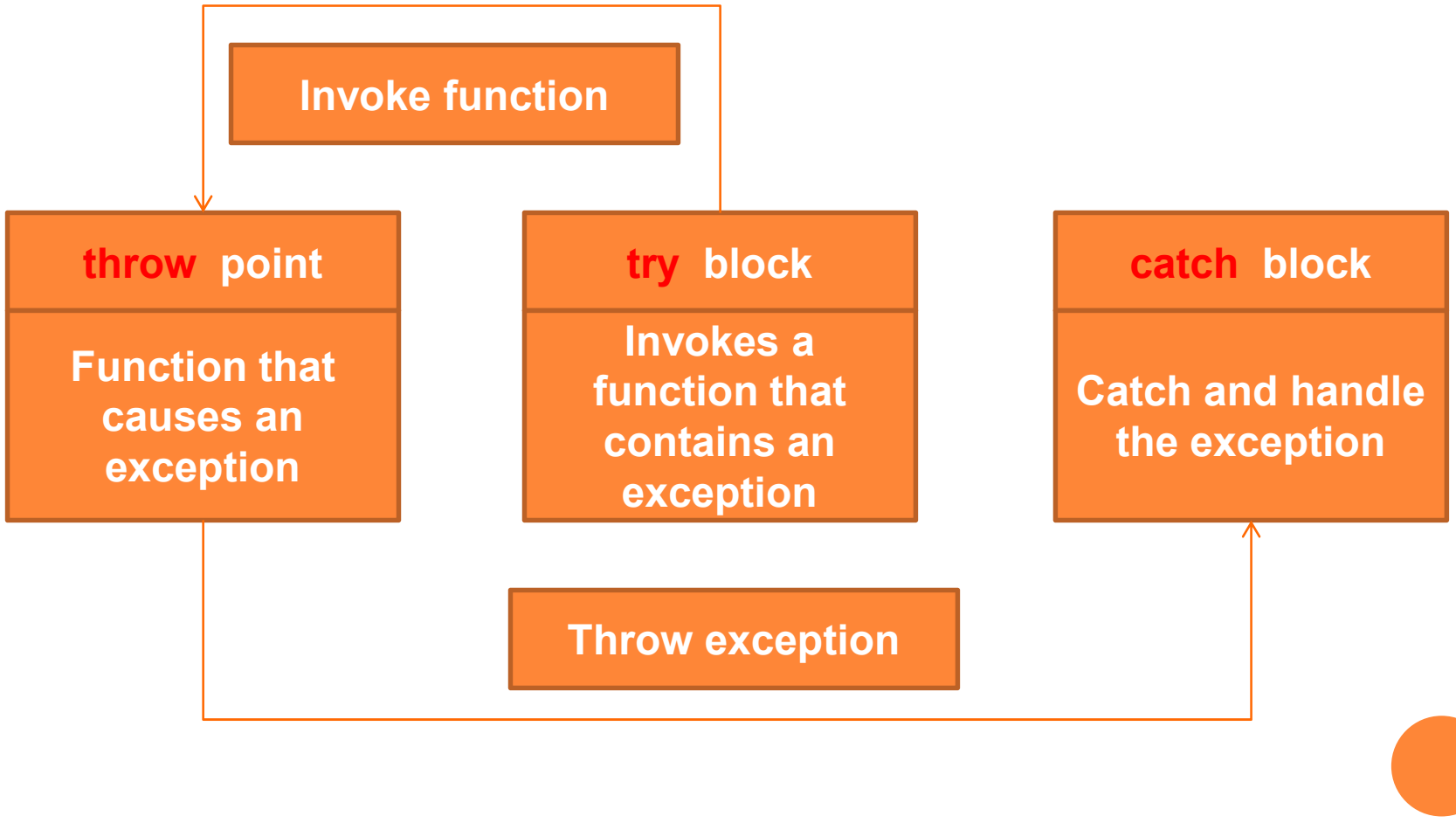
Exception Handling Mechanism (Cont…)

- Exceptions are objects used to transmit information about a problem.
- If the type of the object thrown matches the arg type in the catch statement, the catch block is executed.
- If they do not match, the program is aborted using the **abort()** function (default).
- [E.B.] Program 13.1

Exception Handling Mechanism (Cont…)

- Often, Exceptions are thrown by functions that are invoked from within the try blocks.
- The point at which the throw is executed is called the throw point.
- Once an exception is thrown to the catch block, control cannot return to the throw point.
- [E.B.] Program 13.2

Exception Handling Mechanism (Cont…)



THROWING MECHANISM

- The **throw** statement can have one of the following 3 forms
  - throw(exception)
  - throw exception
  - throw //used to re-throw a exception
- The operand object exception can be of any type, including **constant**.
- It is also possible to throw an object not intended for error handling.

THROWING MECHANISM (CONT…)

- Throw point can be in a deeply nested scope within a try block or in a deeply nested function call.
- In any case, control is transferred to the catch statement.

Catching Mechanism

- The type indicates the type of exception the catch block handles.
- the parameter arg is an **optional** parameter name.
- The catch statement catches an exception whose type matches with the type of the catch argument.

```
catch(type arg)
{
   …
   …
   …
}
```

CATCHING MECHANISM (CONT…)

- If the parameter in the catch statement is named, then the parameter can be used in the exception handling code.
- If a catch statement does not match the exception it is skipped.
- More than one catch statement can be associated with a try block.

CATCHING MECHANISM (CONT…)

```
try
{
        throw exception;
}
catch(type1 arg)
{
        // catch block 1
}
catch(type2 arg)
{
        // catch block 2
}
…
…
catch(typeN arg)
{
        // catch block N
}
```

CATCHING MECHANISM (CONT…)

- When an exception is thrown, the exception handlers are searched **in order** for a match.
- The first handler that yields a match is executed.
- If several catch statement matches the type of an exception the first handler that matches the exception type is executed.
- [E.B.] Program 13.3

Catching Mechanism (Cont…)

- Catch all exception

```
catch (…)
{
        // statement for processing
        // all exception
}
```

- [E.B.] Program 13.4

RETHROWING AN EXCEPTION

- A handler may decide to rethrow the exception caught without processing it.
- In such a case we have to invoke **throw** without any arguments as shown below
  - throw;
- This causes the current exception to be thrown to the next enclosing try/catch sequence and is caught by a catch statement listed after the enclosing try block
- [E.B.] Program 13.5

SPECIFYING EXCEPTION

- It is possible to restrict a function to throw certain specific exceptions by adding a **throw** list clause to the function definition.

```
type function(arg-list) throw(type-list)
{
        … … …
        … … …
        … … …
}
```

- [E.B.] Program 13.6

SPECIFYING EXCEPTION (CONT…)

- The type-list specifies the type of exception that may be thrown.
- Throwing any other kind of exception will cause abnormal program termination.
- If you want to prevent a function from throwing any exception, you may do so by making the type-list empty.

Lecture Contents

- [1] Object Oriented Programming with C++ (3rd Edition) E Balagurusamy
  - Chapter 13 (Full)
- [2]] Teach Yourself C++ (3rd Edition) H Schildt
  - Examples only
- **Study the examples from both books carefully**