Unit 5 - Files and Exception Handling

MCQ OF UNIT-5

Q#1. Which of the following is incorrect about try-with-resources in Java?

(a) try-with-resources was introduced in Java 7
(b) We don't need to use finally block, if we use try-with-resources
(c) The try-with-resources statement ensures that each resource is closed at the end of the statement
(d) It increases the complexity of the code

Answer: d) It increases the complexity of the code

Explanation: The try-with-resources was introduced in Java 7. When we use try-with-resources, the resources will automatically be closed. Hence, we don't need to use finally block. It reduces the complexity of the code and even reduces the lines of code.

Q#2. Which of the following scenarios is best suited for utilizing the try-with-resources statement?

(a) Handling common runtime exceptions
(b) Implementing custom exception classes
(c) Working with IO operations involving streams
(d) Synchronizing multi-threaded operations

Answer: c) Working with IO operations involving streams

Explanation: From the mentioned options, try-with-resources statement will be best suited when working with IO operations involving streams, such as reading from or writing to files. It ensures that the streams are automatically closed after usage, reducing the risk of resource leaks and improving code reliability.

Q#3. What is the output of the following code snippet?

```
try {
    throw new RuntimeException("Error");
} catch (Exception e) {
    System.out.println(e.getMessage());
}
```

(a) Error
(b) RuntimeException
(c) null
(d) The code will not compile.

Answer: a) Error

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

Explanation: The code explicitly throws a RuntimeException with the message "Error". The catch block catches the exception and prints the error message using getMessage().

Q#4. Which of the following statement(s) is/are correct about multi-catch statement?

(a) A single catch block can handle more than one type of exception.
(b) A multi-catch statement is valid in Java 7 and later.
(c) If a catch block handles more than one exception type, then the catch parameter is implicitly final.
(d) Alternatives in a multi-catch statement cannot be related by subclassing.

Answer: (a), (b), (c), (d)

Explanation: All statements are correct about a multi-catch statement since Java 7.

Q#5. Which of the following is the correct syntax for catching multiple exceptions in a single catch block?

(a) catch (ExceptionType1 || ExceptionType2 || ExceptionType3 ex)
(b) catch (ExceptionType1, ExceptionType2, ExceptionType3 ex)
(c) catch (ExceptionType1 && ExceptionType2 && ExceptionType3 ex)
(d) catch (ExceptionType1 | ExceptionType2 | ExceptionType3 ex)

Answer: d) catch (ExceptionType1 | ExceptionType2 | ExceptionType3 e)

Explanation: Multiple exceptions can be caught in a single catch block using the single pipe symbol (|) to separate the exception types.

Q#6. What is wrong with the following code snippet in the context of try-with-resources?

```
static String readFirstLineFromFile() throws IOException {

  try (FileReader fr = new FileReader("");
     BufferedReader br = new BufferedReader(fr)) {
    fr= new FileReader("xyz.txt");
    return br.readLine();
  }
}
```

(a) There is no catch block after the try block.
(b) The re-assignment of variable 'fr' is not allowed.
(c) Two resources can't be declared in a try block.
(d) The throws clause is not needed.

Answer: b) The re-assignment of variable 'fr' is not allowed.

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

Explanation: The resources declared in a try-with-resources context are final by default, hence we can't re-assign them. There will be a compilation error at the same line.

Q#7. Which of the following statements is true about exceptions in the context of Java 7 and later versions?

(a) A try block must be followed by either a catch block or a finally block.
(b) In order to close the resources opened in try block, it is mandatory to include a finally block.
(c) Multiple types of exceptions can be handled by including multiple catch blocks.
(d) It is not mandatory to include a catch block or finally block after a try block.

Answer: d) It is not mandatory to include a catch block or finally block after a try block.

Explanation: Checked exceptions must be caught or declared to be thrown by the method that can potentially throw them.

Q#8. Which exception will be thrown by parseInt() method in Java?

(a) IntegerOutOfBoundException
(b) IntegerFormatException
(c) ArithmeticException
(d) NumberFormatException

Answer: d) NumberFormatException

Explanation: parseInt() method parses input into integer. This method will throw NumberFormatException.

Q#9. Which of the following exception must be either caught or declared to be thrown in Java?

(a) NullPointerException
(b) ArrayIndexOutOfBoundsException
(c) FileNotFoundException
(d) ArithmeticException

Answer: c) FileNotFoundException

Explanation: FileNotFoundException is a checked exception in Java, hence it must be either caught or declared to be thrown.

Q#10. What is the output of the following code snippet?

```
try {
   throw new NullPointerException();
} catch (RuntimeException e) {
   System.out.println("RuntimeException");
} catch (Exception e) {
   System.out.println("Exception");
}
```


Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

(a) RuntimeException
(b) Exception
(c) NullPointerException
(d) The code will not compile.

Answer: a) RuntimeException

Explanation: The code explicitly throws a NullPointerException, which is a subclass of RuntimeException. Since the catch block for RuntimeException is defined first, it is executed.

Q#11. Which of the following is true about the catch block in Java?

(a) A catch block can catch multiple types of exceptions using the semicolon (;).
(b) A catch block can catch multiple types of exceptions using the logical AND operator (&).
(c) A catch block can catch multiple types of exceptions using multiple catch statements.
(d) A catch block can only catch one type of exception at a time.

Answer: c) A catch block can catch multiple types of exceptions using multiple catch statements.

Explanation: Traditionally, multiple types of exceptions can be caught in a catch block by using multiple catch statements, each catching a different exception type. In contrast, a multi-catch statement can catch multiple exceptions in a single catch block since Java 7.

Q#12. Which statement is used to catch and handle multiple exceptions in a single catch block?

(a) catch-all
(b) multi-catch
(c) exception-catch
(d) exception-all

Answer: b) multi-catch

Explanation: The multi-catch statement in Java allows catching and handling multiple exceptions in a single catch block.

Q#13. Which keyword will you use to specify that a method can potentially throw an exception?

(a) try
(b) catch
(c) throw
(d) throws

Answer: d) throws

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

Explanation: We use the throws keyword in a method declaration to state that the method can potentially throw one or more exceptions.

Q#14. What is the purpose of the finally block in exception handling?

(a) To catch and handle exceptions.
(b) To specify that a method can potentially throw an exception.
(c) To execute code regardless of whether an exception is thrown or not.
(d) To explicitly throw an exception.

Answer: c) To execute code regardless of whether an exception is thrown or not.

Explanation: The finally block is used to specify code that should be executed regardless of whether an exception is thrown or not.

Q#15. Which of the following statements is true about the catch block in exception handling?

(a) A try block can have multiple catch blocks.
(b) A catch block can have multiple try blocks.
(c) A catch block must always be followed by a finally block.
(d) A catch block cannot be used without a try block.

Answer: a) A try block can have multiple catch blocks.

Explanation: A try block can have multiple catch blocks to handle different types of exceptions.

Q#16. Which of the following statements is true about the finally block in exception handling?

(a) A finally block is always executed before a catch block.
(b) A finally block is always executed after a catch block.
(c) A finally block is only executed if an exception occurs.
(d) A finally block is optional and can be omitted.

Answer: b) A finally block is always executed after a catch block.

Explanation: A finally block is always executed after a catch block, regardless of whether an exception occurs or not.

Q#17. Which of the following is a subclass of the Exception class?

(a) RuntimeError
(b) Error
(c) Throwable
(d) StackOverflowError

Answer: a) RuntimeError

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

Explanation: RuntimeError is a subclass of the Exception class in Java.

Q#18. Which of the following statements is true about the finally block?

(a) The finally block is required for every try-catch statement.
(b) The finally block is optional and can be omitted.
(c) The finally block is executed only if an exception occurs.
(d) The finally block is executed only if a catch block is present.

Answer: b) The finally block is optional and can be omitted.

Explanation: The finally block is optional and can be omitted in exception handling.

Q#19. What is the output of the following code snippet?

```
try {
   int[] array = new int[5];
   System.out.println(array[5]);
} catch (ArrayIndexOutOfBoundsException e) {
   System.out.println("ArrayIndexOutOfBoundsException");
} finally {
   System.out.println("Finally block executed.");
}
```

(a) ArrayIndexOutOfBoundsException
Finally block executed.
(b) ArrayIndexOutOfBoundsException
(c) Finally block executed.
(d) The code will not compile.

Answer: a) ArrayIndexOutOfBoundsException
Finally block executed.

Explanation: The code attempts to access an element at index 5 in the array which is not available. Hence it raises an ArrayIndexOutOfBoundsException. The catch block is executed, and then the finally block is executed.

Q#20. Which of the following is not a subclass of Throwable in Java?

(a) Checked exception
(b) Unchecked exception
(c) Fatal exception
(d) Error

Answer: c) Fatal exception

Explanation: "Fatal exception" is not a recognized subclass of Throwable in Java. Error is a direct subclass of Throwable.

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

Q#21. Which of the following is/are correct statement(s) about Unchecked Exceptions in Java?

(a) These exceptions occur during the execution of the program.
(b) They are also referred to as Runtime exceptions.
(c) These exceptions are generally ignored during the compilation process.
(d) They are checked while compiling the program.

Answer: (a), (b), (c)

Explanation: They are not checked while compiling the program.

Q#22. Which of the following is an incorrect statement about checked Exceptions in Java?

(a) Checked exceptions are called compile-time exceptions.
(b) They are subtypes of RuntimeException.
(c) These exceptions are checked at compile-time by the compiler.
(d) The IOException is a type of checked Exception.

Answer: b) They are subtypes of RuntimeException

Explanation: They are not subtypes of RuntimeException, but direct subtypes of Exception.

Q#23. In which of the below classes the printStackTrace() method defined?

(a) Exception.
(b) RuntimeException.
(c) Throwable.
(d) Error.

Answer: c) Throwable.

Explanation: The printStackTrace() method is defined in the Throwable class.

Q#24. Which of the following is a not a subclass of the Error class directly or indirectly?

(a) RuntimeError
(b) InternalError
(c) StackOverflowError
(d) OutOfMemoryError

Answer: a) RuntimeError

Explanation: RuntimeError is not a subclass of the Error class directly or indirectly in Java.

Q#25. What is the output of the following code snippet?

```
try {
   throw new Exception("Custom exception");
```


Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

```
} catch (Exception e) {
   System.out.println(e.getMessage());
}
```

(a) Custom exception
(b) Exception
(c) null
(d) The code will not compile.

Answer: a) Custom exception

Explanation: The code explicitly throws an Exception with the message "Custom exception". The catch block catches the exception and prints the error message using getMessage().

Q#26. Which of the following statements is true about custom exception classes in Java?

(a) Custom exception classes must extend the Throwable class.
(b) Custom exception classes must extend the RuntimeException class.
(c) Custom exception classes must extend the Exception class.
(d) Custom exception classes are not required to extend or implement any class or interface.

Answer: a) Custom exception classes must extend the Exception class.

Explanation: In Java, custom exception classes must be subclasses of the Exception class or one of its subclasses.

Q#27. Which of the following is not a type of valid construct in exception handling?

(a) try-catch-finally
(b) try-finally
(c) catch-finally
(d) try-catch

Answer: c) catch-finally

Explanation: "catch-finally" is not a valid exception handling construct in Java. The correct construct is "try-catch" or "try-catch-finally" or "try-finally".

Q#28. Which of the following is not a checked exception in Java?

(a) IOException
(b) FileNotFoundException
(c) NullPointerException
(d) ClassNotFoundException

Answer: c) NullPointerException

Explanation: NullPointerException is an unchecked exception in Java.

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

Q#29. What is the output of the following code snippet?

```
try {
   throw new Error("Fatal error");
} catch (Exception e) {
   System.out.println("Exception");
} catch (Error e) {
   System.out.println("Error");
}
```

(a) Exception
(b) Error
(c) Compiler error
(d) The code will not compile.

Answer: b) Error

Explanation: The code explicitly throws an Error with the message "Fatal error". Since Error is a subclass of Throwable, it matches the catch block for Error, and "Error" is printed.

Q#30. Which of the following exceptions is not a subclass of the RuntimeException class?

(a) NullPointerException
(b) ArrayIndexOutOfBoundsException
(c) IOException
(d) ArithmeticException

Answer: c) IOException

Explanation: IOException is not a subclass of the RuntimeException class. It is a checked exception in Java.

Q#31. Which of the following statements is true about the try-with-resources statement in Java?

(a) It is used to handle multiple exceptions in a single catch block.
(b) It is used to specify that a method can potentially throw an exception.
(c) It is used to automatically close resources after usage.
(d) It is used to define custom exception classes.

Answer: c) It is used to automatically close resources after usage.

Explanation: The try-with-resources statement in Java is used to automatically close resources after usage, ensuring that resources are properly managed and released.

Q#32. Which of the following statements is true about handling exceptions in multi-threaded Java applications?

(a) Each thread should handle exceptions independently.

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

(b) Exceptions thrown by a thread cannot be caught by other threads.
(c) A separate exception handler should be defined for each thread.
(d) Exceptions in multi-threaded applications are handled automatically by the JVM.

Answer: c) A separate exception handler should be defined for each thread.

Explanation: In multi-threaded Java applications, it is recommended to define a separate exception handler for each thread to handle exceptions specific to that thread.

Q#33. What is the output of the following code snippet?

```
try {
   throw new Exception("First Exception");
} catch (Exception e) {
   try {
      throw new Exception("Second Exception");
   } catch (Exception ex) {
      System.out.println(ex.getMessage());
   }
}
```

(a) First Exception
(b) Second Exception
(c) First Exception followed by Second Exception
(d) Second Exception followed by First Exception

Answer: b) Second Exception

Explanation: The code throws the first exception, catches it, and then throws the second exception, which is caught and its message is printed.

Q#34. What is the output of the following code snippet?

```
try {
   throw new Error();
} catch (Throwable t) {
   System.out.println(t.getClass().getSimpleName());
}
```

(a) Error
(b) Throwable
(c) Exception
(d) The code will not compile.

Answer: a) Error

Explanation: The code explicitly throws an Error, which is caught by the catch block. The getClass().getSimpleName() method is used to retrieve the simple name of the caught exception's class.

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

Q#35. Which of the following is the correct syntax for using the try-with-resources statement?

(a) try [Resource r =new Resource()] { // code }
(b) try (Resource r = new Resource(); // code)
(c) try { Resource r = new Resource(); // code }
(d) try (Resource r = new Resource()) // code

Answer: d) try (Resource r = new Resource()) // code

Explanation: Option (d) is the correct syntax.

Q#36. Which of the following interfaces must be implemented by a resource in order to be used with the try-with-resources statement?

(a) Closeable
(b) AutoCloseable
(c) Resource
(d) Disposable

Answer: b) AutoCloseable

Explanation: Resources used with the try-with-resources statement must implement the AutoCloseable interface, which provides the close() method for releasing system resources held by the resource.

Q#37. Which method of AutoCloseable interface is called internally in the try-with-resources statement?

(a) clean()
(b) refresh()
(c) close()
(d) release()

Answer: c) close()

Explanation: The close() method in a resource class is responsible for releasing system resources held by the resource, such as closing file streams or network connections.

Q#38. Which of the following statements is true regarding the order of closing resources in a try-with-resources statement?

(a) Resources are closed in the order of declaration within the try block.
(b) Resources are closed in the reverse order of declaration within the try block.
(c) Resources are closed randomly.
(d) The order of closing resources does not matter.

Answer: b) Resources are closed in the reverse order of declaration within the try block.

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

Explanation: In a try-with-resources statement, resources are closed in the reverse order of their declaration within the try block. This ensures that resources are properly closed, even if an exception occurs.

Q#39. What happens if an exception is thrown both during resource initialization and within the try block of a try-with-resources statement?

(a) The exception thrown during resource initialization takes precedence.
(b) The exception thrown within the try block takes precedence.
(c) Both exceptions are caught and handled.
(d) Only the exception thrown within the try block is caught and handled.

Answer: a) The exception thrown during resource initialization takes precedence.

Explanation: If an exception is thrown both during resource initialization and within the try block, the exception thrown during resource initialization takes precedence. The exception thrown within the try block is added as a suppressed exception.

Q#40. What is the advantage of using the try-with-resources statement instead of a traditional try-catch-finally approach?

(a) It reduces boilerplate code.
(b) It ensures proper resource cleanup without explicitly writing a finally block.
(c) It simplifies complex exception handling.
(d) It improves the performance of exception handling significantly.

Answer: b) It ensures proper resource cleanup without explicitly writing a finally block.

41. What is an exception in C++ program?
a) A problem that arises during the execution of a program
b) A problem that arises during compilation
c) Also known as the syntax error
d) Also known as semantic error
View Answer
Answer: a
Explanation: An exception is defined as the problem in C++ program that arises during the execution of the program for example divide by zero error.

42. By default, what a program does when it detects an exception?
a) Continue running
b) Results in the termination of the program
c) Calls other functions of the program
d) Removes the exception and tells the programmer about an exception
View Answer

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

Answer: b
Explanation: By default, whenever a program detects an exception the program crashes as it does not know how to handle it hence results in the termination of the program.

43. Why do we need to handle exceptions?
a) To avoid unexpected behaviour of a program during run-time
b) To let compiler remove all exceptions by itself
c) To successfully compile the program
d) To get correct output
View Answer
Answer: a
Explanation: We need to handle exceptions in a program to avoid any unexpected behaviour during run-time because that behaviour may affect other parts of the program. Also, an exception is detected during run-time, therefore, a program may compile successfully even with some exceptions cases in your program.

44. How Exception handling is implemented in the C++ program?
a) Using Exception keyword
b) Using try-catch block
c) Using Exception block
d) Using Error handling schedules
View Answer
Answer: b
Explanation: C++ provides a try-catch block to handle exceptions in your program.

45. What is the correct syntax of the try-catch block?
a)

```
try

{

        // programable codes.....

}

catch(Exceptions)

{

        // Code for handling exceptions....

}
```


Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

b)

```
try()
{
        // programable codes.....
}
catch(Exceptions)
{
        // Code for handling exceptions....
}
```

c)

```
try
{
        // programable codes.....
}
catch
{
        // Code for handling exceptions....
}
```

d)

```
try()
{
```

Unit 5 - Files and Exception Handling

```
        // programable codes.....

}

catch

{

        // Code for handling exceptions....

}
```

View Answer

Answer: a

Explanation: Try-catch block has the following syntax:

```
try{
        // codes that needs to check for exceptions
}
catch(Exception E1){
      // codes for handling exception....
      // Exception E denotes the type of exception this block is handling.
}
catch(Exception E2){
        // other exception that needs to be handled...
}
```

You can have any number of catch blocks catching different exceptions.....

46. Which part of the try-catch block is always fully executed?
a) try part
b) catch part
c) finally part
d) throw part

View Answer

47. Which of the following is an exception in C++?
a) Divide by zero
b) Semicolon not written
c) Variable not declared
d) An expression is wrongly written

View Answer

Answer: a

Explanation: Exceptions are those which are encountered during run-time of the

Unit 5 - Files and Exception Handling

program. semicolon, variable not declared and the wrong expression are compile-time errors, therefore, they are not exceptions. Divide by zero is the problem that is encountered during run-time, therefore, it is an exception.

48. What is an error in C++?
a) Violation of syntactic and semantic rules of a languages
b) Missing of Semicolon
c) Missing of double quotes
d) Violation of program interface
View Answer
Answer: a
Explanation: An error occurs when rules and laws of a language is violated while writing programs in that language.

49. What is the difference between error and exception?
a) Both are the same
b) Errors can be handled at the run-time but the exceptions cannot
c) Exceptions can be handled at the run-time but the errors cannot
d) Both can be handled during run-time
View Answer
Answer: c
Explanation: Exceptions can be handled during run-time whereas errors cannot be because exceptions occur due to some unexpected conditions during run-time whereas about errors compiler is sure and tells about them during compile-time.

50. What are the different types of exceptions?
a) 1
b) 2
c) 3
d) 4
View Answer
Answer: b
Explanation: There are two types of exceptions: Synchronous and asynchronous exceptions. Synchronous exceptions that are caused by the event which can be controlled by the program whereas Asynchronous exceptions are those which are beyond the control of the program.

51. Which keyword is used to throw an exception?
a) try
b) throw
c) throws
d) except
View Answer

Unit 5 - Files and Exception Handling

Answer: b
Explanation: 'throw' keyword is used to throw exceptions if something bad happens.

52. What will be the output of the following C++ code?

```
#include <iostream>
#include <string>
#include <cstdlib>
using namespace std;

void func(int a, int b)
{

        if(b == 0){
                throw "This value of b will make the product zero. "
                   "So please provide positive values.\n";
        }
        else{
                cout<<"Product of "<<a<<" and  "<<b<<" is: "<<a*b<<endl;
        }
}

int main()
{
        try{
                func(5,0);
        }
        catch(const char* e){
                cout<<e;
        }
}
```

a) 0
b) 5
c) This value of b will make the product zero. So please provide positive values.
d) Product of 5 and 0 is: 0
View Answer
Answer: c
Explanation: As the value of b = 0 is provided to the func() and the function is throwing an exception whenever the value of b = 0. Therefore the function throws the exception which will be printed on the screen.
Output:

```
$ ./a.out

This value of b will make the product zero. So please provide positive values.
```


Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

53. What will be the output of the following C++ code?

```
#include <iostream>
#include <string>
#include <cstdlib>
using namespace std;
void func(int a, int b)
{
        if(b == 0){
                throw "This value of b will make the product zero. "
                    "So please provide positive values.\n";
        }
        else{
                cout<<"Product of "<<a<<" and  "<<b<<" is: "<<a*b<<endl;
        }
}

int main()
{
        try{
                func(5,0);
        }
        catch(char* e){
                cout<<e;
        }
}
```

a) 0
b) Aborted (core dumped)
c) This value of b will make the product zero. So please provide positive values.
d) Product of 5 and 0 is: 0

View Answer

Answer: b

Explanation: As the func() is throwing a const char* string but we the catch block is not catching any const char* exception i.e. exception thrown is not handled therefore the program results into Aborted(core dumped).

Output:

```
$ ./a.out

terminate called after throwing an instance of 'char const*'

Aborted (core dumped)
```

54. What is Re-throwing an exception means in C++?

a) An exception that is thrown again as it is not handled by that catching block

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

b) An exception that is caught twice
c) An exception that is not handled in one caught hence thrown again
d) All of the mentioned
View Answer
55. What will be the output of the following C++ code?

```
#include <iostream>
#include <string>
#include <cstdlib>
using namespace std;
void func(int a, int b)
{
          if(b < 1){
                    throw b;
          }
          else{
                    cout<<"Product of "<<a<<" and  "<<b<<" is: "<<a*b<<endl;
          }
}

int main()
{
          try
         {
                    try
                {
                     func(5,-1);
                    }
                    catch(int b)
                {
                              if(b==0)
                                        throw "value of b is zero\n";
                              else
                                        throw "value of b is less than zero\n";
                    }
          }
          catch(const char* e)
         {
                    cout<<e;
          }

}
```


Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

a) value of b is zero
b) value of b is less than zero
c) Product of 5 and -1 is: -5
d) Aborted(core dumped)
View Answer
Answer: b
Explanation: Here the func() throws the value of b which is caught by the inner try-catch block, which again throws the message inorder to handle different cases of b which is caught by the outer try-catch block. Now as the value of b is negative the program outputs the message as shown.
Output:

```
$ ./a.out

value of b is less than zero
```

56. Where should we place catch block of the derived class in a try-catch block?
a) Before the catch block of Base class
b) After the catch block of Base class
c) Anywhere in the sequence of catch blocks
d) After all the catch blocks
View Answer
Answer: a
Explanation: C++ asks the programmer to place the catch block of derived class before a catch block of the base class, otherwise derived catch block will never be executed.

57. What happens when this C++ program is compiled?

```
#include <iostream>
#include <string>
#include <cstdlib>
using namespace std;
class A
{
        int a;
    public:
        A(){}
};

class B: public A
{
        int b;
    public:
        B(){}
```

Unit 5 - Files and Exception Handling

```
};

void func()
{
        B b;
        throw b;
}

int main()
{
        try{
                func();
        }
        catch(A a){
                cout<<"Caught A Class\n";
        }
        catch(B b){
                cout<<"Caught B Class\n";
        }
}
```

a) The program compiles successfully without any errors or warnings
b) Compile-time error occurs
c) The program compiles successfully with warnings
d) The program gives both errors and warnings
View Answer
Answer: c
Explanation: Catch block of derived should always be placed before the catch block base class, hence the program gives warnings stating that exceptions of the derived class will be caught by the base class.

58. What will be the output of the following C++ code?

```
#include <iostream>
#include <string>
#include <cstdlib>
using namespace std;
class A
{
        int a;
    public:
        A(){}
};
```


Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

```
class B: public A
{
        int b;
    public:
        B(){}
};

void func()
{
        B b;
        throw b;
}

int main()
{
        try{
                func();
        }
        catch(A a){
                cout<<"Caught A Class\n";
        }
        catch(B b){
                cout<<"Caught B Class\n";
        }
}
```

a) Caught B Class
b) Caught A Class
c) Compile-time error
d) Run-time error

View Answer

Answer: b

Explanation: As the catch block of the derived class is after the catch block of base class, therefore, all the exceptions of the derived class will be caught by the base class, Hence the output of catch block of class A is printed.

59. What will be the output of the following C++ code?

```
#include <iostream>
#include <string>
#include <cstdlib>
using namespace std;
class A
{
        int a;
```


Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

```
   public:
         A(){}
};

class B: public A
{
         int b;
    public:
         B(){}
};

void func()
{
         B b;
         throw b;
}

int main()
{
         try{
                  func();
         }
         catch(B b){
                  cout<<"Caught B Class\n";
         }
         catch(A a){
                  cout<<"Caught A Class\n";
         }
}
```

a) Caught B Class
b) Caught A Class
c) Compile-time error
d) Run-time error

View Answer

Answer: a

Explanation: In this as the catch block of the derived class is before the catch block of the base class so when func() throws the object of class B it is caught by the catch block of class B, Hence the output is printed as shown.

Output:

```
$ ./a.out

Caught B Class
```

61. What will be the output of the following C++ code?

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

```
#include <iostream>
#include <string>
#include <cstdlib>
using namespace std;
class A
{
        int a;
    public:
        A(){}
};
class B: public A
{
        int b;
    public:
        B(){}
};

void func()
{
        B b;
        throw b;
}
int main()
{
        try{
                func();
        }
        catch(B *b){
                cout<<"Caught B Class\n";
        }
        catch(A a){
                cout<<"Caught A Class\n";
        }
}
```

a) Caught B Class
b) Caught A Class
c) Compile-time error
d) Run-time error

View Answer

Answer: b

Explanation: The func() throws the object of class B but as catch block is defined to catch the exception of class B, Therefore the exception is caught by the base class A. The programmer has defined the catch block for B*, therefore, the object B is not caught by the pointer object B*.

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

62. What id the syntax for catching any type of exceptions?
a) catch(Exception e)
b) catch(...)
c) catch(Exception ALL)
d) catch(ALL)
View Answer
Answer: b
Explanation: catch(...) is used in C++ to catch all types of exceptions in a single catch block.

63. What will be the output of the following C++ code?

```
#include <iostream>
#include <string>
#include <cstdlib>
using namespace std;
class A
{
          int a;
    public:
          A(){}
};

class B: public A
{
          int b;
    public:
          B(){}
};

void func1()
{
          B b;
          throw b;
}

void func2()
{
          A a;
          throw a;
}

int main()
{
```


Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

```
        try{
                func1();
        }
        catch(...){
                cout<<"Caught All types of exceptions\n";
        }
        try{
                func2();
        }
        catch(B b){
                cout<<"Caught All types of exceptions\n";
        }
}
```

a) Caught All types of exceptions
b)

Caught All types of exceptions

Aborted (core dumped)

c) Error
d)

Caught All types of exceptions

Caught All types of exceptions

View Answer
Answer: b
Explanation: Two try-catch blocks is declared each catching the respective exceptions from class A and B. But as we have defined catch all exceptions in the first case, therefore, the exception for class B is caught when thrown by the func1(), but in the second case, the try-catch block is catching only the exception for class B so when func2() throws class A exception and no catch block to catch that exception therefore program results into abort(core dumped).

64. Uncaught exception leads to _____
a) termination of program
b) successful execution of programs
c) no effect on the program

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

d) execution of other functions of the program starts
View Answer
Answer: a
Explanation: Uncaught exceptions in a program leads to the termination of a program.

65. An uncaught handler returns to _____
a) main function
b) its caller
c) its callee
d) none of the mentioned
View Answer
Answer: d
Explanation: Uncaught exceptions do not “return” to any specific location in the program. They trigger a chain of events leading to program termination. In C++, when an uncaught exception occurs, the program unwinds the stack to find an appropriate exception handler. It searches backward through the call stack (i.e., the series of function calls that led to the point where the exception occurred) until it finds a function that has a suitable exception handler. If no handler is found, the program may terminate or exhibit undefined behavior.

66. Header file used for exception handling in C++?
a) <cstdlib>
b) <string>
c) <handler>
d) <exception>
View Answer
Answer: d
Explanation: <exception> header file is used to use exception handler in C++.

. The C++ code which causes abnormal termination/behaviour of a program should be written under _____ block.
a) try
b) catch
c) finally
d) throw
View Answer
Answer: a
Explanation: Code that leads to the abnormal termination of the program should be written under the try block.

67. Exception handlers are declared with _____ keyword.
a) try
b) catch

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

c) throw
d) finally
View Answer
Answer: b
Explanation: C++ uses catch block to handle any exceptions that occur during run-time of the program.

68. Which of the following statements are correct about Catch handler?

i. It must be placed immediately after the try block

ii. It can have more than one parameters

iii. There must be one and only one catch handler for every try block

iv. There can be multiple catch handler for a try block

v. General catch handler can be kept anywhere after try block.

a) i, iv, v
b) i, ii, iii
c) i, iv
d) i, ii
View Answer
Answer: c
Explanation: A catch block should always be placed after the try block and there can be multiple catch block following a try block.

68. In nested try-catch block, if the inner catch block gets executed, then_____
a) Program stops immediately
b) Outer catch block also executes
c) Compiler jumps to the outer catch block and executes remaining statements of the main() function
d) Compiler executes remaining statements of outer try-catch block and then the main() function
View Answer
Answer: d
Explanation: The inner catch block will be executed then remaining part of the outer try block will be executed and then the main bock will be executed.

68. If inner catch block is unable to handle the exception thrown then_____
a) The compiler looks for the outer try-catch block

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

b) Program stops abnormally
c) The compiler will check for appropriate catch handler of the outer try block
d) The compiler will not check for appropriate catch handler of the outer try block
View Answer
Answer: c
Explanation: In such cases, the compiler will try to find an appropriate outer catch block to handle the exception otherwise if nothing is there then occurs the abnormal behaviour of the program.

69. In nested try catch blocks, if both inner and outer catch blocks are unable to handle the exception thrown, then _____
a) Compiler executes only main()
b) Compiler throws compile time errors about it
c) Program will run without any interrupt
d) Program will be termianted abnormally
View Answer
Answer: d
Explanation: If no inner/outer catch handler is avaliable to handle the exception then as usual the program will show abnormal behaviour.

70. Which function is invoked when an unhandled exception is thrown?
a) stop()
b) aborted()
c) terminate()
d) abandon()
View Answer
Answer: c
Explanation: terminate() function is called/invoked incase any exception is not handled properly.

71. How one can restrict a function to throw particular exceptions only?
a) By defining multiple try-catch blocks inside a function
b) By defining a generic function within a try-catch block
c) By defining a function with throw clauses
d) Not allowed in C++
View Answer
Answer: c
Explanation: We can use throw clause to mention the exceptions that a function can throw. Hence restricting the function to throw some particular exceptions only.

72. Which function is invoked when we try to throw an exception that is not supported by a function?
a) indeterminate()
b) unutilized()

Unit 5 - Files and Exception Handling

c) unexpected()
d) unpredicted()
View Answer
Answer: c
Explanation: As the exception is not supported by the function so it does not know what to do about the exception in that case it call the unexpected() function of the STL library.

73. Return type of uncaught_exception() is_____
a) int
b) bool
c) char *
d) double
View Answer
Answer: b
Explanation: Return type of uncaught exceptions are bool.

74. What will be the output of the following C++ code?

```
#include <iostream>
using namespace std;
int main()
{
   int var = -12;
   try {
      cout<<"Inside try\n";
      if (var < 0)
      {
        throw var;
        cout<<"After throw\n";
      }
   }
   catch (int var ) {
      cout<<"Exception Caught\n";
   }

   cout<<"After catch\n";
   return 0;
}
```

a)

Inside try

Exception Caught

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

After catch

b)

Inside try

After throw

After catch

c)

Inside try

Exception Caught

After throw

d)

Inside try

Exception Caught

After throw

After catch

View Answer

Answer: a

Explanation: “Inside try” will always be printed as we just entering try block then. Now as var < 0 therefore the try block will throw int var as exception hence “After throw” will not be printed) Now this exception will be caught by the catch handler printing “Exception caught” and at last after terminating the program “After catch” will be printed.

75. What will be the output of the following C++ code?

```
#include <iostream>
using namespace std;
```


Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

```
int main()
{
   int var = -12;
   try {
      cout<<"Inside try\n";
      if (var < 0)
      {
        throw var;
        cout<<"After throw\n";
      }
   }
   catch (char var ) {
      cout<<"Exception Caught\n";
   }

   cout<<"After catch\n";
   return 0;
}
```

a)

Inside try

Exception Caught

After catch

b)

Inside try

After throw

After catch

c) Error
d) Run-time error
View Answer
Answer: d
Explanation: As no catch handler is defined to catch an integer hence when var variable, which is int, is thrown then nothing is there to catch the int hence the program terminates abnormally.

76. What will be the output of the following C++ code?

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

```
#include <iostream>
using namespace std;
int main()
{
    try
    {
        try
        {
            throw 20;
        }
        catch (int n)
        {
            cout << "Inner Catch\n";
        }
    }
    catch (int x)
    {
        cout << "Outer Catch\n";
    }
    return 0;
}
```

a) Inner Catch
b) Outer Catch
c)

Inner Catch

Outer Catch

d) Error
View Answer
Answer: a
Explanation: As exception thrown by the inner try block is caught by the inner catch block hence the exception is handled at the inner level and program continues to run outer try block, statement afterwards.

77. What will be the output of the following C++ code?

```
#include <iostream>
using namespace std;
int main()
{
    try
```


Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

```
    {
        try
        {
            throw 20;
        }
        catch (char n)
        {
            cout << "Inner Catch\n";
        }
    }
    catch (int x)
    {
        cout << "Outer Catch\n";
    }
    return 0;
}
```

a) Inner Catch
b) Outer Catch
c)

Inner Catch

Outer Catch

d) Error
View Answer
Answer: b
Explanation: As there is no inner catch handler to handle the int exception thrown by the try block therefore outer catch block handler catches the exception thrown by the inner try catch therefore the output prints “Outer Catch” instead of “Inner Catch”. After that program continues execution.

78. What will be the output of the following C++ code?

```
#include <iostream>
using namespace std;
int main()
{
    try
    {
        try
        {
            throw 20;
```


Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

```
        }
        catch (int n)
        {
            cout << "Inner Catch\n";
            throw;
        }
    }
    catch (int x)
    {
        cout << "Outer Catch\n";
    }
    return 0;
}
```

a) Inner Catch
b) Outer Catch
c)

Inner Catch

Outer Catch

d) Error
View Answer
Answer: c
Explanation: The exception thrown by the inner try catch block is caught by the inner block hence “Inner Catch” is printed but as inner catch block again throws an exception further therefore the exception is thrown further which is caught by the outer catch block hence “Outer Catch” is also printed.

79. Which of the following is true about exception handling in C++?

i) There is a standard exception class in C++ similar to Exception class in Java.

ii) All exceptions are unchecked in C++, i.e., the compiler does not checks if the exceptions are caught or not.

iii) In C++, a function can specify the list of exceptions that it can throw using comma separated list like following.

void fun(int a, char b) throw (Exception1, Exception2, ..)

a) i, iii
b) i, ii, iii

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

c) i, ii
d) ii, iii
View Answer
Answer: b
Explanation: In C++ also we have an exception class similar to java. All exceptions are unchecked in C++. We can specify the list of exception that a function throws using the above format.

80. Which header file is required to use file I/O operations?
a) <ifstream>
b) <ostream>
c) <fstream>
d) <iostream>
View Answer
Answer: c
Explanation: <fstream> header file is needed to use file I/O operations in C++. This header file contains all the file I/O operations definition.

81. Which of the following is used to create an output stream?
a) ofstream
b) ifstream
c) iostream
d) fsstream
View Answer
Answer: a
Explanation: ofstream is used to create an output stream in C++ file handling operations. Ofstream objects are used to read files.

82. Which of the following is used to create a stream that performs both input and output operations?
a) ofstream
b) ifstream
c) iostream
d) fstream
View Answer
Answer: d
Explanation: fstream is used to create a stream that performs both input and output operations in C++ file handling.

83. Which of the following is not used as a file opening mode?
a) ios::trunc
b) ios::binary
c) ios::in

Unit 5 - Files and Exception Handling

d) ios::ate
View Answer
Answer: a
Explanation: ios::trunc is used to truncate a file if it exists. It is not a file opening mode.

84. Which of the following statements are correct?

1) It is not possible to combine two or more file opening mode in open() method.

2) It is possible to combine two or more file opening mode in open() method.

3) ios::in and ios::out are input and output file opening mode respectively.

a) 1, 3
b) 2, 3
c) 3 only
d) 1, 2
View Answer
Answer: b
Explanation: C++ allows to use one or more file opening mode in a single open() method. ios::in and ios::out are input and output file opening mode respectively.

85. By default, all the files in C++ are opened in _____ mode.
a) Text
b) Binary
c) ISCII
d) VTC
View Answer
Answer: a
Explanation: By default, all the files in C++ are opened in text mode. They read the file as normal text.

86. What is the use of ios::trunc mode?
a) To open a file in input mode
b) To open a file in output mode
c) To truncate an existing file to half
d) To truncate an existing file to zero
View Answer
Answer: d
Explanation: In C++ file handling, ios::trunc mode is used to truncate an existing file to zero length.

Unit 5 - Files and Exception Handling

87. Which of the following is the default mode of the opening using the ofstream class?
a) ios::in
b) ios::out
c) ios::app
d) ios::trunc
View Answer
Answer: b
Explanation: By default, the file is opened in ios::out mode if the file object we are using is of ofstream class.

88. What is the return type open() method?
a) int
b) char
c) bool
d) float
View Answer
Answer: c
Explanation: open() method returns a bool value indicating whether the file is opened or some error has occurred.

89. Which of the following is not used to seek file pointer?
a) ios::set
b) ios::end
c) ios::cur
d) ios::beg
View Answer
Answer: a
Explanation: ios::set is not used to seek file pointer. ios::end is used to seek from the end of the file. ios::curr from the current position. ios::beg from the beginning.

90. Which of the following is the default mode of the opening using the ifstream class?
a) ios::in
b) ios::out
c) ios::app
d) ios::trunc
View Answer
Answer: a
Explanation: By default, the file is opened in ios::in mode if the file object we are using is of ifstream class.

91. Which of the following is the default mode of the opening using the fstream class?

Unit 5 - Files and Exception Handling

a) ios::in
b) ios::out
c) ios::in|ios::out
d) ios::trunc
View Answer
Answer: c
Explanation: By default, the file is opened in ios::in|ios::out mode if the file object we are using is of fstream class.

92. Which function is used in C++ to get the current position of file pointer in a file?
a) tell_p()
b) get_pos()
c) get_p()
d) tell_pos()
View Answer
Answer: a
Explanation: C++ provides tell_p() function to get the current position of the file pointer in a file.

93. Which function is used to reposition the file pointer?
a) moveg()
b) seekg()
c) changep()
d) go_p()
View Answer
Answer: b
Explanation: seekg() function is used to reposition a file pointer in a file. The function takes the offset and relative position from where we need to shift out pointer.

94. Which of the following is used to move the file pointer to start of a file?
a) ios::beg
b) ios::start
c) ios::cur
d) ios::first
View Answer
Answer: a
Explanation: ios::beg is used to reposition the file pointer to the beginning of the file. It is whenever you want to reposition the pointer at the beginning from any point to the start of the file.

96. Which operator is used to signify the namespace?
a) conditional operator
b) ternary operator

Unit 5 - Files and Exception Handling

c) scope operator
d) bitwise operator
View Answer
Answer: c
Explanation: Scope operator(::) is used in namespace syntax.
General syntax:
namespace X{ int a;}
cout<<X::a;

97. Identify the correct statement.
a) Namespace is used to group class, objects and functions
b) Namespace is used to mark the beginning of the program
c) A namespace is used to separate the class, objects
d) Namespace is used to mark the beginning & end of the program
View Answer
Answer: a
Explanation: Namespace allows you to group class, objects, and functions. It is used to divide the global scope into the sub-scopes.

98. What is the use of Namespace?
a) To encapsulate the data
b) To structure a program into logical units
c) Encapsulate the data & structure a program into logical units
d) It is used to mark the beginning of the program
View Answer
Answer: b
Explanation: The main aim of the namespace is to understand the logical units of the program and to make the program so robust.

99. What is the general syntax for accessing the namespace variable?
a) namespace::operator
b) namespace,operator
c) namespace#operator
d) namespace$operator
View Answer
Answer: a
Explanation: To access variables from namespace we use following syntax.
namespace :: variable;
General syntax:
namespace X{ int a;}
cout<<X::a;

100. What will be the output of the following C++ code?

Unit 5 - Files and Exception Handling

```
1.  #include <iostream>
2.  using namespace std;
3.  namespace first
4.  {
5.      int var = 5;
6.  }
7.  namespace second
8.  {
9.      double var = 3.1416;
10.  }
11.  int main ()
12.  {
13.      int a;
14.      a = first::var + second::var;
15.      cout << a;
16.      return 0;
17.  }
```

a) 8.31416
b) 8
c) 9
d) compile time error
View Answer
Answer: b
Explanation: As we are getting two variables from namespace variable and we are adding that.
Output:

```
$ g++ name.cpp
$ a.out
8
```

101. What will be the output of the following C++ code?

```
1.  #include <iostream>
2.  using namespace std;
3.  namespace first
4.  {
5.      int x = 5;
6.      int y = 10;
7.  }
8.  namespace second
9.  {
10.      double x = 3.1416;
11.      double y = 2.7183;
12.  }
```


Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

```
13.    int main ()
14.    {
15.        using first::x;
16.        using second::y;
17.        bool a, b;
18.        a = x > y;
19.        b = first::y < second::x;
20.        cout << a << b;
21.        return 0;
22.    }
```

a) 11
b) 01
c) 00
d) 10

View Answer

Answer: d

Explanation: We are inter mixing the variable and comparing it which is bigger and smaller and according to that we are printing the output.

Output:

```
$ g++ name1.cpp
$ a.out
10
```

102. What will be the output of the following C++ code?

```
1.    #include <iostream>
2.    using namespace std;
3.    namespace Box1
4.    {
5.        int a = 4;
6.    }
7.    namespace Box2
8.    {
9.        int a = 13;
10.    }
11.    int main ()
12.    {
13.        int a = 16;
14.        Box1::a;
15.        Box2::a;
16.        cout << a;
17.        return 0;
18.    }
```


Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

a) 4
b) 13
c) 16
d) compile time error
View Answer
Answer: c
Explanation: In this program, as there is lot of variable a and it is printing the value inside the block because it got the highest priority.
Output:

```
$ g++ name2.cpp
$ a.out
16
```

103. What will be the output of the following C++ code?

```
1.    #include <iostream>
2.    using namespace std;
3.    namespace space
4.    {
5.        int x = 10;
6.    }
7.    namespace space
8.    {
9.        int y = 15;
10.    }
11.    int main(int argc, char * argv[])
12.    {
13.        space::x = space::y =5;
14.        cout << space::x << space::y;
15.    }
```

a) 1015
b) 1510
c) 55
d) compile time error
View Answer
Answer: c
Explanation: We are overriding the value at the main function and so we are getting the output as 55.
Output:

```
$ g++ name4.cpp
$ a.out
55
```

104. What will be the output of the following C++ code?

Unit 5 - Files and Exception Handling

```
1.  #include <iostream>
2.  using namespace std;
3.  namespace extra
4.  {
5.      int i;
6.  }
7.  void i()
8.  {
9.      using namespace extra;
10.     int i;
11.     i = 9;
12.     cout << i;
13.  }
14.  int main()
15.  {
16.     enum  letter { i, j};
17.     class i { letter j; };
18.     ::i();
19.     return 0;
20.  }
```

a) 9
b) 10
c) compile time error
d) 11
View Answer
Answer: a
Explanation: A scope resolution operator without a scope qualifier refers to the global namespace.

106. Which keyword is used to access the variable in the namespace?
a) using
b) dynamic
c) const
d) static
View Answer
Answer: a
Explanation: using keyword is used to specify the name of the namespace to which the variable belongs.
eg.
namespace A{ int a = 5;}
namespace B{ int a = 10;}
using namespace A;
cout<<a; // will print value of a from namespace A.

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

using namespace B;
cout<<a; // will print value of a from namespace B.

107. Pick the incorrect statement for namespaces in C++.
a) Namespace declarations are always global scope
b) Keyword namespace is used at the starting of a namespace definition
c) Namespace has access specifiers like private or public
d) Namespace definitions can be nested
View Answer
Answer: c
Explanation: Namespace does not have any specifiers associated with it like classes or structures.

108. Which operator is used for accessing a member of namespace?
a) :
b) ::
c) ->
d) .
View Answer
Answer: b
Explanation: Scope resolution operator(::) is used for accessing a member of a namespace. example:

```
namespace A{
        int var;
}
A::var = 5;
```

3. Which is the correct syntax of declaring a namespace?
a)

namespace A{

int i

}

b)

namespace B{

int i;

};

Faculty: SHAHRUKH KAMAL
Shahrukhkamal7@gmail.com

Unit 5 - Files and Exception Handling

c)

```
namespace C{
        int i;
}
```

d)

```
Namespace D{
        int i
}
```

View Answer

Answer: c

Explanation: A namespace definition always starts with the namespace keyword so definition with Namespace(capital N) is wrong. namespace does is not terminated by a semicolon hence the definition with a semicolon is wrong. every variable declaration in C++ should end with semicolon therefore namespace containing 'int i' without semicolon is wrong.