|  |  |
| --- | --- |
| **1.** | public class M1 {  public static void main(String[] args) {  int i = 10 / 0;  int j = Integer.parseInt("abc");  String s1 = null;  int k = s1.length();  int[] elements = new int[5];  int m = elements[200];  Object obj1 = new Integer(90);  String str = (String) obj1;  main(null);  int[] array = new int[999999999];  }  } |
| |  | | --- | | A.  NumberFormatException |  |  | | --- | | B.  NullPointerException |  |  | | --- | | C.  ClassCastException |  |  | | --- | | D.  ArithmeticException | | | |
|  | | |
| **2.** | public class M2 {  public static void main(String[] args) {  //Class.forName("");    try  {  Class.forName("");  }  catch(ClassNotFoundException ex)  {  ex.printStackTrace();  }  }  } |
| |  | | --- | | A.  ArithmeticException |  |  | | --- | | B.  ClassNotFoundException |  |  | | --- | | C.  ClassCastException |  |  | | --- | | D.  NullPointerException | | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **3.** | import java.io.FileReader;  import java.io.IOException;  public class M4 {  public static void main(String[] args) {      //FileReader f1 = new FileReader("");      try  {  FileReader f2 = new FileReader("");  }  catch(IOException ex)  {    }  }  } | |
| |  | | --- | | A.  ArithmeticException |  |  | | --- | | B.  ClassNotFoundException |  |  | | --- | | C.  ClassCastException |  |  | | --- | | D.  NullPointerException |  |  | | --- | | E.  None of the above | | | | |
|  | | | |
| **4.** | import java.sql.DriverManager;  import java.sql.SQLException;  public class M5 {  public static void main(String[] args) {    //DriverManager.getConnection("");    try  {  DriverManager.getConnection("");  }  catch(SQLException ex)  {    }  }  } |
| |  | | --- | | A.  ArithmeticException |  |  | | --- | | B.  ClassNotFoundException |  |  | | --- | | C.  ClassCastException |  |  | | --- | | D.  No output |  |  | | --- | | E.  None of the above | | | | |
|  | | | |

|  |  |
| --- | --- |
| **5.** | public class M6 {  public static void main(String[] args) {      //Thread.sleep(10000);      try  {  Thread.sleep(10000);  }  catch(InterruptedException ex)  {    }  }  } |
| |  | | --- | | A.  ArithmeticException |  |  | | --- | | B.  ClassNotFoundException |  |  | | --- | | C.  ClassCastException |  |  | | --- | | D.  No output |  |  | | --- | | E.  None of the above | | | |
|  | | |
| **6.** | import java.text.ParseException;  import java.text.SimpleDateFormat;  public class M7 {  public static void main(String[] args) {  SimpleDateFormat sd = null;    //sd.parse("");      try  {  sd.parse("");  }  catch(ParseException ex)  {    }    }  } |
| |  | | --- | | A.  ArithmeticException |  |  | | --- | | B.  ClassNotFoundException |  |  | | --- | | C.  NullPointerException |  |  | | --- | | D.  ClassCastException |  |  | | --- | | E.  None of the above | | | |
|  | | |

|  |  |
| --- | --- |
| **7.** | package com.lara2;  public class M1 {  public static void main(String[] args) {  System.out.println("main begin");  System.out.println("-------");  System.out.println("---------");  String s1 = null;  System.out.println("---------");  Object obj1 = new Integer(40);  System.out.println("main end");  }  } |
| |  | | --- | | A.  main begin  -------  ---------  ---------  main end |  |  | | --- | | B.  NullPointerException |  |  | | --- | | C.  Compile Time Error | | | |
|  | | |
| **8.** | package com.lara2;  public class M2 {  public static void main(String[] args) {  System.out.println("main begin");  System.out.println("-------");  int[] elements = new int[999999999];  System.out.println("-------");  System.out.println("main end");  }  } |
| |  | | --- | | A.  main begin  -------  ---------  ---------  main end |  |  | | --- | | B.  NullPointerException |  |  | | --- | | C.  ArrayIndexOutOfBoundsException |  |  | | --- | | D.  StackOverFlowError |  |  | | --- | | E.  none | | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **9.** | package com.lara2;  public class M3 {  public static void main(String[] args) {  System.out.println("main begin");  try  {  System.out.println(1);  main(null);  System.out.println(2);  }  catch(StackOverflowError err)  {  System.out.println(3);  }  System.out.println("main end");  }  }  /\* find out error occured is getting handled or not\*/ | |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | | |
|  | | | |
| **10.** | | package com.lara2;  public class M4 {  public static void main(String[] args) {  System.out.println("main begin");  try  {  System.out.println(1);  int[] elements = new int[999999999];  System.out.println(2);  }  catch(OutOfMemoryError err)  {  System.out.println(3);  }  System.out.println("main end");  }  } |
| |  | | --- | | A.  OutOfMemoryError |  |  | | --- | | B.  main begin  1  3  main end |  |  | | --- | | C.  ArrayIndexOutOfBoundsException | | | | |
|  | | | |

|  |  |
| --- | --- |
| **11.** | package com.lara1;  public class M1 {  int test1()  {  return 10;  }      int test2(boolean flag)  {  return 10;  }    int test3(boolean flag)  {  if(flag)  {  return 10;  }  return 20;  }    int test4(boolean flag)  {  if(flag)  {  return 10;  }  else  {  return 20;  }  }  int test7(boolean flag)  {  if(flag)  {  return 20;  }  else  {    }  return 10;  }    int test8(boolean flag)  {  if(flag)  {    }  else  {  return 20;  }  return 10;  }  }  //compilation success or not? |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |
| **12.** | package com.lara1;  public class M2 {  int test1()  {  try  {  //some statements  }  catch(ArithmeticException ex)  {  //some statements  }  return 10;  }      int test2()  {  try  {  //some statements  return 20;  }  catch(ArithmeticException ex)  {  //some statements  }  return 10;  } |
| |  | | --- | | A.  no |  |  | | --- | | B.  yes | | | |
|  | | |

|  |  |
| --- | --- |
| **13.** | package com.lara1;  public class M3 {  int test1()  {  try  {  //some statements  }  catch(ArithmeticException ex)  {  //some statements  }  catch(NullPointerException ex)  {  //some statements  }  return 10;  }      int test2()  {  try  {  //some statements  return 10;  }  catch(ArithmeticException ex)  {  //some statements  }  catch(NullPointerException ex)  {  //some statements  }  return 20;  }  int test3()  {  try  {  //some statements  }  catch(ArithmeticException ex)  {  //some statements  return 10;  }  catch(NullPointerException ex)  {  //some statements  }  return 20;  }      int test4()  {  try  {  //some statements  }  catch(ArithmeticException ex)  {  //some statements  return 10;  }  catch(NullPointerException ex)  {  //some statements  return 5;  }  return 20;  }  int test5()  {  try  {  //some statements  }  catch(ArithmeticException ex)  {  //some statements  }  catch(NullPointerException ex)  {  //some statements  return 5;  }  return 20;  }  } |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |
| **14.** | package com.lara1;  public class M4 {  int test1()  {  try  {  //some statements  }  catch(ArithmeticException ex)  {  //some statements  }  finally  {  //some statments  }  return 10;  }    int test2()  {  try  {  //some statements  return 20;  }  catch(ArithmeticException ex)  {  //some statements  }  finally  {  //some statments  }  return 10;  }    int test3()  {  try  {  //some statements  }  catch(ArithmeticException ex)  {  //some statements  return 20;  }  finally  {  //some statments  }  return 10;  }    int test4()  {  try  {  //some statements  }  catch(ArithmeticException ex)  {  //some statements  }  finally  {  //some statments  return 20;  }  } |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |

|  |  |
| --- | --- |
| **15.** | class H {  void test1() throws NullPointerException  {    }  }  class I extends H {  void test2() throws ClassCastException  {    }  }  /\* whether it compiles successfully\*/ |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |
| **16.** | public class AgeIsNegativeException extends ArithmeticException  {  public AgeIsNegativeException()  {    }  public AgeIsNegativeException(String msg)  {  super(msg);  }  }  /\* whether it compiles successfully\*/ |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |

|  |  |
| --- | --- |
| **17.** | public class B {  public static void main(String[] args) {  System.out.println(1);  if(true)  {  throw new ArithmeticException();  }  System.out.println(2);  }  }  /\* whether it compiles successfully\*/ |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |
| **18.** | public class C {  public static void main(String[] args) {  System.out.println(1);  if(true)  {  throw new ArithmeticException("some message");  }  System.out.println(2);  }  }  /\* whether it compiles successfully\*/ |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **19.** | import java.util.Scanner;  public class D {  public static void main(String[] args) {  System.out.println(1);  Scanner sc = new Scanner(System.in);  System.out.println("enter age");  int age = sc.nextInt();  if(age <= 0)  {  throw new ArithmeticException("age should be greater than 0");  }  System.out.println(2);  }  }  /\* if you give input as  -23  then what could be the output\*/ | |
| |  | | --- | | A.  1  enter age  -23  2 |  |  | | --- | | B.  1  enter age  -23  Exception in thread "main" java.lang.ArithmeticException: age should be greater than 0  at com.lara5.D.main(D.java:13) |  |  | | --- | | C.  Compile Time Error | | | | |
|  | | | |
| **20.** | import java.util.Scanner;  public class E {  public static void main(String[] args) {  System.out.println(1);  Scanner sc = new Scanner(System.in);  System.out.println("enter age");  int age = sc.nextInt();  if(age <= 0)  {  throw new AgeIsNegativeException("age should be greater than 0");  }  //continue  System.out.println(2);  }  }  class AgeIsNegativeException extends ArithmeticException  {  public AgeIsNegativeException()  {    }  public AgeIsNegativeException(String msg)  {  super(msg);  }  }  /\* if you give input as  64  then what could be the output\*/ |
| |  | | --- | | A.  1  enter age  64  2 |  |  | | --- | | B.  1  enter age  64  Exception in thread "main" java.lang.ArithmeticException: age should be greater than 0  at com.lara5.D.main(D.java:13) |  |  | | --- | | C.  Compile Time Error | | | | |
|  | | | |

|  |  |
| --- | --- |
| **21.** | public class F {  public static void main(String[] args) {  System.out.println(1);  try  {  System.out.println(2);  if(true)  {  throw new ArithmeticException();  }  System.out.println(3);  }  catch(ArithmeticException ex)  {  System.out.println(4);  }  System.out.println(5);  }  }  /\* Whether it compiles successfully or not?\*/ |
| |  | | --- | | A.  yes |  |  | | --- | | B.  No | | | |
|  | | |
| **22.** | public class G {  public static void main(String[] args) {  System.out.println(1);  try  {  System.out.println(2);  int i = 10 / 0;  System.out.println(3);  }  catch(ArithmeticException ex)  {  System.out.println(4);  throw new ArithmeticException(ex.getMessage());  }  System.out.println(5);  }  } |
| |  | | --- | | A.  1  2  4  ArthmeticException: / by zero |  |  | | --- | | B.  1  2  4  Exception in thread "main" java.lang.ArithmeticException: / by zero  at com.lara5.G.main(G.java:15) |  |  | | --- | | C.  1  2 | | | |
|  | | |

|  |  |
| --- | --- |
| **23.** | class H {  void test() throws RuntimeException  {    }  }  class I extends H {  void test() throws Exception  {    }  } |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |
| **24.** | class H {  void test()  {    }  }  class I extends H {  void test() throws IOException  {    }  } |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |

|  |  |
| --- | --- |
| **25.** | class H {  void test() throws Throwable  {    }  }  class I extends H {  void test() throws Exception  {    }  } |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |
| **26.** | class H {  void test() throws RuntimeException  {    }  }  class I extends H {  void test() throws NullPointerException  {    }  } |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |

|  |  |
| --- | --- |
| **27.** | public class M1 {  public static void main(String[] args) throws ClassNotFoundException{  Class.forName("");  }  }  /\* Whether it is compilation successfull are not?\*/ |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |
| **28.** | public class M2 {  public static void main(String[] args) {  try  {  test();  }  catch(ClassNotFoundException ex)  {    }  }  static void test() throws ClassNotFoundException  {  Class.forName("");  }  }  /\* Whether it is compilation successfull are not?\*/ |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |

|  |  |
| --- | --- |
| **29.** | public class M3 {  public static void main(String[] args) throws ClassNotFoundException{  test();  }  static void test() throws ClassNotFoundException  {  Class.forName("");  }  }  /\* Whether it is compilation successfull are not?\*/ |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |
| **30.** | public class M4 {  public static void main(String[] args) throws ClassNotFoundException {  test1();  }  static void test1() throws ClassNotFoundException  {  test2();  }  static void test2() throws ClassNotFoundException  {  Class.forName("");  }  }  /\* Whether it is compilation successfull are not?\*/ |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |

|  |  |
| --- | --- |
| **31.** | public class M5 {  public static void main(String[] args) {  test1();  }  static void test1()  {  test2();  }  static void test2()  {  int i = 10 / 0;  }  }  /\* Whether it is compilation successfull are not?\*/ |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |
| **32.** | public class M6 {  public static void main(String[] args) throws ArithmeticException{  test1();  }  static void test1() throws ArithmeticException  {  test2();  }  static void test2() throws ArithmeticException  {  int i = 10 / 0;  }  }  /\* Whether it is compilation successfull are not?\*/ |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |

|  |  |
| --- | --- |
| **33.** | import java.sql.DriverManager;  public class M7 {  public static void main(String[] args) throws Exception {  test1();  }  static void test1() throws Exception  {  test2();  }  static void test2() throws Exception  {  Class.forName("");  DriverManager.getConnection("");  Thread.sleep(230);  }  }  /\* Whether it is compilation successfull are not?\*/ |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |
| **34.** | import java.sql.DriverManager;  import java.sql.SQLException;  public class M8 {  public static void main(String[] args){  test1();  }  static void test1()  {  test2();  }  static void test2()  {  try  {  Class.forName("");  DriverManager.getConnection("");  Thread.sleep(230);  }  catch(SQLException ex)  {    }  catch(ClassNotFoundException ex)  {    }  catch(InterruptedException ex)  {    }  }  }  /\* Whether it is compilation successfull are not?\*/ |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |

|  |  |
| --- | --- |
| **35.** | import java.sql.DriverManager;  import java.sql.SQLException;  public class M9 {  public static void main(String[] args){  test1();  }  static void test1()  {  test2();  }  static void test2()  {  try  {  Class.forName("");  DriverManager.getConnection("");  Thread.sleep(230);  }  catch(Exception ex)  {    }  }  }  /\* Whether it is compilation successfull are not?\*/ |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | | |
|  | | | |
| **36.** | import java.sql.DriverManager;  import java.sql.SQLException;  public class M10 {  public static void main(String[] args){  test1();  }  static void test1()  {  try  {  test2();  }  catch(ClassNotFoundException ex)  {    }  catch(SQLException ex)  {    }  catch(InterruptedException ex)  {    }  }  static void test2() throws ClassNotFoundException, SQLException, InterruptedException  {  Class.forName("");  DriverManager.getConnection("");  Thread.sleep(230);  }  }  /\* Whether it is compilation successfull are not?\*/ | |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | | |
|  | | | |

|  |  |
| --- | --- |
| **37.** | import java.sql.DriverManager;  import java.sql.SQLException;  public class M11 {  public static void main(String[] args){  test1();  }  static void test1()  {  try  {  test2();  }  catch(Exception ex)  {    }  }  static void test2() throws ClassNotFoundException, SQLException, InterruptedException  {  Class.forName("");  DriverManager.getConnection("");  Thread.sleep(230);  }  }  /\* Whether it is compilation successfull are not?\*/ |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |
| **38.** | import java.sql.DriverManager;  import java.sql.SQLException;  public class M12 {  public static void main(String[] args)  throws ClassNotFoundException, SQLException, InterruptedException  {  test1();    }  static void test1() throws ClassNotFoundException, SQLException, InterruptedException  {  test1();  }  static void test2() throws ClassNotFoundException, SQLException, InterruptedException  {  Class.forName("");  DriverManager.getConnection("");  Thread.sleep(230);  }  }  /\* Whether it is compilation successfull are not?\*/ |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |

|  |  |
| --- | --- |
| **39.** | public class M13 {  public static void main(String[] args) {  try  {  System.out.println("done");  int i = 10;  i++;  i += 2000;  }  catch(ArithmeticException ex)  {    }  }  }  /\* Whether it is compilation successfull are not?\*/ |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |
| **40.** | public class M14 {  public static void main(String[] args) {  try  {  System.out.println("done");  int i = 10;  i++;  i += 2000;  }  catch(ClassNotFoundException ex)  {    }  }  }  /\* Whether it is compilation successfull are not?\*/ |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |

|  |  |
| --- | --- |
| **41.** | package com.lara;  import java.util.Scanner;  public class M26 {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  System.out.println("enter some thing");  String s1 = sc.next();  try  {  System.out.println(1);  int i = Integer.parseInt(s1);  System.out.println(2);  int k = i / (i - 9);  System.out.println(3);  }  catch(ArithmeticException ex)  {  System.out.println(4);  }  catch(NumberFormatException ex)  {  System.out.println(5);  }  System.out.println(6);  }  }  /\* If you enter  main  what could be the output\*/ |
| |  | | --- | | A.  1  NumberFormatException |  |  | | --- | | B.  1  2  3  5 |  |  | | --- | | C.  Compile Time Error |  |  | | --- | | D.  None | | | |
|  | | |
| **42.** | package com.lara;  import java.util.Scanner;  public class M27 {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  System.out.println("enter some thing");  String s1 = sc.next();  try  {  System.out.println(1);  int i = Integer.parseInt(s1);  System.out.println(2);  int k = i / (i - 9);  System.out.println(3);  }  catch(ArithmeticException ex)  {  System.out.println(4);  }  System.out.println(6);  }  }  /\*if you enter  123589  then what could be the output |
| |  | | --- | | A.  1  NumberFormatException |  |  | | --- | | B.  1  AithmeticException |  |  | | --- | | C.  Compile Time Error |  |  | | --- | | D.  None | | | |
|  | | |

|  |  |
| --- | --- |
| **43.** | package com.lara;  import java.util.Scanner;  public class M28 {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  System.out.println("enter some thing");  String s1 = sc.next();  try  {  System.out.println(1);  int i = Integer.parseInt(s1);  System.out.println(2);  int k = i / (i - 9); ////risky 2  System.out.println(3);  int[] elements = new int[10];  int m = elements[i];  System.out.println(4);  }  catch(ArithmeticException ex)  {  System.out.println(5);  }  catch(NumberFormatException ex1)  {  System.out.println(6);  }  catch(ArrayIndexOutOfBoundsException ex1)  {  System.out.println(7);  }  System.out.println(8);  }  }  /\* if you enter  abcg12345  then what could be the output/\* |
| |  | | --- | | A.  1  6  8 |  |  | | --- | | B.  1  2  3  7  8 |  |  | | --- | | C.  Compile Time Error |  |  | | --- | | D.  None | | | |
|  | | |
| **44.** | package com.lara;  public class M29 {  public static void main(String[] args) {  try  {    }  catch(NumberFormatException ex)  {    }    try  {    }  catch(NumberFormatException ex)  {    }  }  } |
| |  | | --- | | A.  NumberFormatException |  |  | | --- | | B.  NullPionterException |  |  | | --- | | C.  Compile Time Error |  |  | | --- | | D.  No output | | | |
|  | | |

|  |  |
| --- | --- |
| **45.** | package com.lara;  public class M30 {  public static void main(String[] args) {  try  {    }  catch(ArrayIndexOutOfBoundsException ex)  {    }  catch(NullPointerException ex)  {    }  }  } |
| |  | | --- | | A.  ArrayOutOfBoundsException |  |  | | --- | | B.  NullPionterException |  |  | | --- | | C.  Compile Time Error |  |  | | --- | | D.  No output | | | |
|  | | |
| **46.** | package com.lara;  import java.util.Scanner;  public class M31 {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  System.out.println("enter some thing");  String s1 = sc.next();  try  {  System.out.println(1);  int i = Integer.parseInt(s1);  System.out.println(2);  int k = i / (i - 9);  System.out.println(3);  }  catch(NumberFormatException ex)  {  System.out.println(4);  }  finally  {  System.out.println(5);  }  System.out.println(6);  }  }  /\* if you enter  9  then what could be the output\*/ |
| |  | | --- | | A.  1  2  3  5  6 |  |  | | --- | | B.  1  4  5  6 |  |  | | --- | | C.  1  2  5  AirthmeticException |  |  | | --- | | D.  None | | | |
|  | | |

|  |  |
| --- | --- |
| **47.** | package com.lara;  public class M32 {  public static void main(String[] args) {  System.out.println(1);  int i = 10 / 0;  try  {  System.out.println(2);  }  catch(ArithmeticException ex)  {  System.out.println(3);  }  finally  {  System.out.println(4);  }  }  } |
| |  | | --- | | A.  1  4 |  |  | | --- | | B.  1  AithhmeticException |  |  | | --- | | C.  Compile Time Error |  |  | | --- | | D.  None | | | |
|  | | |
| **48.** | package com.lara;  public class M33 {  public static void main(String[] args) {  System.out.println(1);  try  {  System.out.println(2);  return;  }  catch(ArithmeticException ex)  {  System.out.println(3);  }  finally  {  System.out.println(4);  }  System.out.println(5);  }  } |
| |  | | --- | | A.  1  2 |  |  | | --- | | B.  1  2  5 |  |  | | --- | | C.  1  2  4  5 |  |  | | --- | | D.  1  2  4 |  |  | | --- | | E.  None | | | |
|  | | |

|  |  |
| --- | --- |
| **49.** | package com.lara;  public class M34 {  public static void main(String[] args) {  System.out.println(1);  try  {  System.out.println(2);  int i = 10 / 0;  }  catch(ArithmeticException ex)  {  System.out.println(3);  return;  }  finally  {  System.out.println(4);  }  System.out.println(5);  }  } |
| |  | | --- | | A.  1  2  3  4 |  |  | | --- | | B.  1  2  3 |  |  | | --- | | C.  1  2  3  4 |  |  | | --- | | D.  none | | | |
|  | | |
| **50.** | package com.lara;  public class M35 {  public static void main(String[] args) {  System.out.println(1);  if(true)  {  return;  }  try  {  System.out.println(2);  }  catch(ArithmeticException ex)  {  System.out.println(3);  }  finally  {  System.out.println(4);  }  System.out.println(5);  }  } |
| |  | | --- | | A.  1  4 |  |  | | --- | | B.  1  2  4  5 |  |  | | --- | | C.  1 |  |  | | --- | | D.  none | | | |
|  | | |

|  |  |
| --- | --- |
| **51.** | package com.lara;  public class M36 {  public static void main(String[] args) {  System.out.println(1);  try  {  System.out.println(2);  System.exit(0);  }  catch(ArithmeticException ex)  {  System.out.println(3);  }  finally  {  System.out.println(4);  }  System.out.println(5);  }  } |
| |  | | --- | | A.  Compile Time Error |  |  | | --- | | B.  1  2 |  |  | | --- | | C.  1  2  4 |  |  | | --- | | D.  none | | | |
|  | | |
| **52.** | package com.lara;  public class M37 {  public static void main(String[] args) {  System.out.println(1);  try  {  System.out.println(1);  int i = 10 / 0;  }  catch(ArithmeticException ex)  {  System.out.println(3);  System.exit(0);  }  finally  {  System.out.println(4);  }  System.out.println(5);  }  } |
| |  | | --- | | A.  1  1  3 |  |  | | --- | | B.  1  1  3  4 |  |  | | --- | | C.  1  1  3  5 |  |  | | --- | | D.  none | | | |
|  | | |

|  |  |
| --- | --- |
| **53.** | package com.lara;  public class M1 {  public static void main(String[] args) {  System.out.println("main begin");  int i = 10 / 0;  System.out.println("main end");  }  } |
| |  | | --- | | A.  main begin  main end |  |  | | --- | | B.  Compile Time Error |  |  | | --- | | C.  Run Time Exception |  |  | | --- | | D.  None | | | |
|  | | |
| **54.** | package com.lara;  public class M1 {  public static void main(String[] args) {  System.out.println("main begin");  int i = 10 / 0;  System.out.println("main end");  }  } |
| |  | | --- | | A.  main begin  main end |  |  | | --- | | B.  Compile Time Error |  |  | | --- | | C.  ArithmeticException |  |  | | --- | | D.  None | | | |
|  | | |

|  |  |
| --- | --- |
| **55.** | package com.lara;  public class M2 {  public static void main(String[] args) {  System.out.println("main begin");  int i = Integer.parseInt("90R");  System.out.println("main end");  }  } |
| |  | | --- | | A.  main begin  main end |  |  | | --- | | B.  NumberFormatException |  |  | | --- | | C.  ArithmeticException |  |  | | --- | | D.  Compile Time Error |  |  | | --- | | E.  None | | | |
|  | | |
| **56.** | package com.lara;  public class M3 {  public static void main(String[] args) {  System.out.println("main begin");  String s1 = null;  int i = s1.length();  System.out.println("main end");  }  } |
| |  | | --- | | A.  main begin  main end |  |  | | --- | | B.  NumberFormatException |  |  | | --- | | C.  Compile Time Error |  |  | | --- | | D.  NullPointerException |  |  | | --- | | E.  None | | | |
|  | | |

|  |  |
| --- | --- |
| **57.** | package com.lara;  public class M4 {  public static void main(String[] args) {  System.out.println("main begin");  int[] elements = new int[5];  int i = elements[10];  System.out.println("main end");  }  } |
| |  | | --- | | A.  main begin  main end |  |  | | --- | | B.  NumberFormatException |  |  | | --- | | C.  Compile Time Error |  |  | | --- | | D.  NullPointerException |  |  | | --- | | E.  None | | | |
|  | | |
| **58.** | package com.lara;  public class M5 {  public static void main(String[] args) {  System.out.println("main begin");  Object obj = new Object();  String s1 = (String) obj;  System.out.println("main end");  }  } |
| |  | | --- | | A.  main begin  main end |  |  | | --- | | B.  NumberFormatException |  |  | | --- | | C.  ClassCastException |  |  | | --- | | D.  NullPointerException |  |  | | --- | | E.  None | | | |
|  | | |

|  |  |
| --- | --- |
| **59.** | package com.lara;  public class M6 {  public static void main(String[] args) {  System.out.println("main begin");  Object obj = "abc";  String s1 = (String) obj;  System.out.println("main end");  }  } |
| |  | | --- | | A.  main begin  main end |  |  | | --- | | B.  NumberFormatException |  |  | | --- | | C.  ClassCastException |  |  | | --- | | D.  NullPointerException |  |  | | --- | | E.  None | | | |
|  | | |
| **60.** | package com.lara;  public class M7 {  public static void main(String[] args) {  System.out.println("main begin");  Object obj = new Integer("90");  String s1 = (String) obj;  System.out.println("main end");  }  } |
| |  | | --- | | A.  main begin  main end |  |  | | --- | | B.  NumberFormatException |  |  | | --- | | C.  ClassCastException |  |  | | --- | | D.  NullPointerException |  |  | | --- | | E.  None | | | |
|  | | |

|  |  |
| --- | --- |
| **61.** | package com.lara;  public class M8 {  public static void main(String[] args) {  System.out.println("main begin");  String s1 = "hello";  String s2 = s1.substring(2, 10);  System.out.println("main end");  }  } |
| |  | | --- | | A.  main begin  main end |  |  | | --- | | B.  StringIndexOutOfBoundsException |  |  | | --- | | C.  ClassCastException |  |  | | --- | | D.  NullPointerException |  |  | | --- | | E.  None | | | |
|  | | |
| **62.** | package com.lara;  public class M9 {  public static void main(String[] args) {  System.out.println("main begin");  int[] elements = new int[999999999];  System.out.println("main end");  }  } |
| |  | | --- | | A.  main begin  main end |  |  | | --- | | B.  ArrayIndexOutOfBoundsException |  |  | | --- | | C.  OutOfMemoryError |  |  | | --- | | D.  NullPointerException |  |  | | --- | | E.  None | | | |
|  | | |

|  |  |
| --- | --- |
| **63.** | package com.lara;  public class M10 {  public static void main(String[] args) {  System.out.println("main begin");  main(null);  System.out.println("main end");  }  } |
| |  | | --- | | A.  main begin  main end |  |  | | --- | | B.  StackOverFlowError |  |  | | --- | | C.  OutOfMemoryError |  |  | | --- | | D.  NullPointerException |  |  | | --- | | E.  None | | | |
|  | | |
| **64.** | package com.lara;  public class M11 {  public static void main(String[] args) {  System.out.println("main begin");  main(args);  System.out.println("main end");  }  } |
| |  | | --- | | A.  main begin  main end |  |  | | --- | | B.  StackOverFlowError |  |  | | --- | | C.  OutOfMemoryError |  |  | | --- | | D.  NullPointerException |  |  | | --- | | E.  None | | | |
|  | | |

|  |  |
| --- | --- |
| **65.** | package com.lara;  public class M12 {  public static void main(String[] args) {  System.out.println("main begin");  try  {  System.out.println(1);  int i = 10 / 0;  System.out.println(2);  }  catch(ArithmeticException ex)  {  System.out.println(3);  }  System.out.println("main end");  }  } |
| |  | | --- | | A.  main begin  1  3  main end |  |  | | --- | | B.  ArithmeticException |  |  | | --- | | C.  NumberFormatException |  |  | | --- | | D.  NullPointerException |  |  | | --- | | E.  None | | | |
|  | | |
| **66.** | package com.lara;  public class M13 {  public static void main(String[] args) {  System.out.println("main begin");  try  {  System.out.println(1);  int i = 10 / 0;  System.out.println(2);  }  catch(ArithmeticException ex)  {  System.out.println("from catch:" + ex);  }  System.out.println("main end");  }  } |
| |  | | --- | | A.  main begin  1  from catch : / by zero  main end |  |  | | --- | | B.  ArithmeticException |  |  | | --- | | C.  NumberFormatException |  |  | | --- | | D.  main begin  1  from catch:java.lang.ArithmeticException: / by zero  main end |  |  | | --- | | E.  None | | | |
|  | | |

|  |  |
| --- | --- |
| **67.** | package com.lara;  public class M13 {  public static void main(String[] args) {  System.out.println("main begin");  try  {  System.out.println(1);  int i = 10 / 0;  System.out.println(2);  }  catch(ArithmeticException ex)  {  System.out.println("from catch:" + ex);  }  System.out.println("main end");  }  } |
| |  | | --- | | A.  main begin  1  from catch : / by zero  main end |  |  | | --- | | B.  ArithmeticException |  |  | | --- | | C.  NumberFormatException |  |  | | --- | | D.  main begin  1  from catch:java.lang.ArithmeticException: / by zero  main end |  |  | | --- | | E.  None | | | |
|  | | |
| **68.** | What will be the output of the program?  public class Foo  {  public static void main(String[] args)  {  try  {  return;  }  finally  {  System.out.println( "Finally" );  }  }  } |
| |  | | --- | | A.  finally |  |  | | --- | | B.  compilation fail |  |  | | --- | | C.  the code runn with no output |  |  | | --- | | D.  An exception is thrown at runtime. | | | |
|  | | |

|  |  |
| --- | --- |
| **69.** | What will be the output of the program?  try  {  int x = 0;  int y = 5 / x;  }  catch (Exception e)  {  System.out.println("Exception");  }  catch (ArithmeticException ae)  {  System.out.println(" Arithmetic Exception");  }  System.out.println("finished"); |
| |  | | --- | | A.  finished |  |  | | --- | | B.  Exception |  |  | | --- | | C.  Compilation fail |  |  | | --- | | D.  Arithmetic Exception | | | |
|  | | |
| **70.** | What will be the output of the program?  ublic class X  {  public static void main(String [] args)  {  try  {  badMethod();  System.out.print("A");  }  catch (Exception ex)  {  System.out.print("B");  }  finally  {  System.out.print("C");  }  System.out.print("D");  }  public static void badMethod()  {  throw new Error();  } |
| |  | | --- | | A.  ABCD |  |  | | --- | | B.  compilation fail |  |  | | --- | | C.  C is printing before exiting with an error message. |  |  | | --- | | D.  BC is printed before exiting with an error message. | | | |
|  | | |

|  |  |
| --- | --- |
| **71.** | What will be the output of the program?  public class X  {  public static void main(String [] args)  {  try  {  badMethod();  System.out.print("A");  }  catch (RuntimeException ex)  {  System.out.print("B");  }  catch (Exception ex1)  {  System.out.print("C");  }  finally  {  System.out.print("D");  }  System.out.print("E");  }  public static void badMethod()  {  throw new RuntimeException();  }  } |
| |  | | --- | | A.  BD |  |  | | --- | | B.  BCD |  |  | | --- | | C.  BDE |  |  | | --- | | D.  BCDE | | | |
|  | | |
| **72.** | What will be the output of the program?  public class RTExcept  {  public static void throwit ()  {  System.out.print("throwit ");  throw new RuntimeException();  }  public static void main(String [] args)  {  try  {  System.out.print("hello ");  throwit();  }  catch (Exception re )  {  System.out.print("caught ");  }  finally  {  System.out.print("finally ");  }  System.out.println("after ");  }  } |
| |  | | --- | | A.  hello throwit caught |  |  | | --- | | B.  hello throwit caught |  |  | | --- | | C.  hello throwit RuntimeException caught after |  |  | | --- | | D.  hello throwit caught finally after | | | |
|  | | |

|  |  |
| --- | --- |
| **73.** | What will be the output of the program?  public class Test  {  public static void aMethod() throws Exception  {  try  {  throw new Exception();  }  finally  {  System.out.print("finally ");  }  }  public static void main(String args[])  {  try  {  aMethod();  }  catch (Exception e)  {  System.out.print("exception ");  }  System.out.print("finished");  }  } |
| |  | | --- | | A.  finally |  |  | | --- | | B.  exception finished |  |  | | --- | | C.  finally exception finished |  |  | | --- | | D.  Compilation fails | | | |
|  | | |
| **74.** | What will be the output of the program?  public class X  {  public static void main(String [] args)  {  try  {  badMethod();  System.out.print("A");  }  catch (Exception ex)  {  System.out.print("B");  }  finally  {  System.out.print("C");  }  System.out.print("D");  }  public static void badMethod() {}  } |
| |  | | --- | | A.  AC |  |  | | --- | | B.  BC |  |  | | --- | | C.  ACD |  |  | | --- | | D.  ABCD | | | |
|  | | |

|  |  |
| --- | --- |
| **75.** | What will be the output of the program?  public class X  {  public static void main(String [] args)  {  try  {  badMethod();  System.out.print("A");  }  catch (Exception ex)  {  System.out.print("B");  }  finally  {  System.out.print("C");  }  System.out.print("D");  }  public static void badMethod()  {  throw new RuntimeException();  }  } |
| |  | | --- | | A.  AB |  |  | | --- | | B.  BC |  |  | | --- | | C.  ABC |  |  | | --- | | D.  BCD | | | |
|  | | |
| **76.** | What will be the output of the program?  public class MyProgram  {  public static void main(String args[])  {  try  {  System.out.print("Hello world ");  }  finally  {  System.out.println("Finally executing ");  }  }  } |
| |  | | --- | | A.  Nothing. The program will not compile because no exceptions are specified. |  |  | | --- | | B.  Nothing. The program will not compile because no catch clauses are specified. |  |  | | --- | | C.  Hello world. |  |  | | --- | | D.  Hello world Finally executing | | | |
|  | | |

|  |  |
| --- | --- |
| **77.** | What will be the output of the program  class Exc0 extends Exception { }  class Exc1 extends Exc0 { }  public class Test  {  public static void main(String args[])  {  try  {  throw new Exc1();  }  catch (Exc0 e0)  {  System.out.println("Ex0 caught");  }  catch (Exception e)  {  System.out.println("exception caught");  }  }  } |
| |  | | --- | | A.  Ex0 caught |  |  | | --- | | B.  exception caught |  |  | | --- | | C.  Compilation fails because of an error at line 2. |  |  | | --- | | D.  Compilation fails because of an error at line 9. | | | |
|  | | |
| **78.** | Predict the output of following Java program  class Main {  public static void main(String args[]) {  try {  throw 10;  }  catch(int e) {  System.out.println("Got the Exception " + e);  }  }  } |
| |  | | --- | | A.  Got the Exception 10 |  |  | | --- | | B.  Got The Exception 0 |  |  | | --- | | C.  Compilation Fail |  |  | | --- | | D.  None | | | |
|  | | |

|  |  |
| --- | --- |
| **79.** | class Test extends Exception { }    class Main {  public static void main(String args[]) {  try {  throw new Test();  }  catch(Test t) {  System.out.println("Got the Test Exception");  }  finally {  System.out.println("Inside finally block ");  }  }  } |
| |  | | --- | | A.  Got the Test Exception  Inside finally block |  |  | | --- | | B.  Got the Test Exception |  |  | | --- | | C.  Inside finally block |  |  | | --- | | D.  Compiler Error | | | |
|  | | |
| **80.** | Output of following Java program?  class Main {  public static void main(String args[]) {  int x = 0;  int y = 10;  int z = y/x;  }  } |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  compiles and runs fine |  |  | | --- | | C.  Compiles fine but throws ArithmeticException | | | |
|  | | |

|  |  |
| --- | --- |
| **81.** | class Base extends Exception {}  class Derived extends Base {}    public class Main {  public static void main(String args[]) {  // some other stuff  try {  // Some monitored code  throw new Derived();  }  catch(Base b) {  System.out.println("Caught base class exception");  }  catch(Derived d) {  System.out.println("Caught derived class exception");  }  }  } |
| |  | | --- | | A.  Caught base class exception |  |  | | --- | | B.  Caught derived class exception |  |  | | --- | | C.  Compiler Error because derived is not throwable |  |  | | --- | | D.  Compiler Error because base class exception is caught before derived class | | | |
|  | | |
| **82.** | class Test  {  public static void main (String[] args)  {  try  {  int a = 0;  System.out.println ("a = " + a);  int b = 20 / a;  System.out.println ("b = " + b);  }    catch(ArithmeticException e)  {  System.out.println ("Divide by zero error");  }    finally  {  System.out.println ("inside the finally block");  }  }  } |
| |  | | --- | | A.  Compile error |  |  | | --- | | B.  Divide by zero error |  |  | | --- | | C.  a = 0  Divide by zero error  inside the finally bloc |  |  | | --- | | D.  a = 0 |  |  | | --- | | E.  inside the finally block | | | |
|  | | |

|  |  |
| --- | --- |
| **83.** | class Test  {  public static void main(String[] args)  {  try  {  int a[]= {1, 2, 3, 4};  for (int i = 1; i <= 4; i++)  {  System.out.println ("a[" + i + "]=" + a[i] + "n");  }  }    catch (Exception e)  {  System.out.println ("error = " + e);  }    catch (ArrayIndexOutOfBoundsException e)  {  System.out.println ("ArrayIndexOutOfBoundsException");  }  }  } |
| |  | | --- | | A.  Compiler error |  |  | | --- | | B.  Run time error |  |  | | --- | | C.  ArrayIndexOutOfBoundsException |  |  | | --- | | D.  Error Code is printed |  |  | | --- | | E.  Array is printed | | | |
|  | | |
| **84.** | class Test  {  String str = "a";    void A()  {  try  {  str +="b";  B();  }  catch (Exception e)  {  str += "c";  }  }    void B() throws Exception  {  try  {  str += "d";  C();  }  catch(Exception e)  {  throw new Exception();  }  finally  {  str += "e";  }    str += "f";    }    void C() throws Exception  {  throw new Exception();  }    void display()  {  System.out.println(str);  }    public static void main(String[] args)  {  Test object = new Test();  object.A();  object.display();  }    } |
| |  | | --- | | A.  abdef |  |  | | --- | | B.  abdec |  |  | | --- | | C.  abdefc | | | |
|  | | |

|  |  |
| --- | --- |
| **85.** | Predict the output of the following program.  class Test  { int count = 0;    void A() throws Exception  {  try  {  count++;    try  {  count++;    try  {  count++;  throw new Exception();    }    catch(Exception ex)  {  count++;  throw new Exception();  }  }    catch(Exception ex)  {  count++;  }  }    catch(Exception ex)  {  count++;  }    }    void display()  {  System.out.println(count);  }    public static void main(String[] args) throws Exception  {  Test obj = new Test();  obj.A();  obj.display();  }  } |
| |  | | --- | | A.  4 |  |  | | --- | | B.  5 |  |  | | --- | | C.  6 |  |  | | --- | | D.  Compilation fail | | | |
|  | | |
| **86.** | What is the output of this program?  class exception\_handling  {  public static void main(String args[])  {  try  {  int a, b;  b = 0;  a = 5 / b;  System.out.print("A");  }  catch(ArithmeticException e)  {  System.out.print("B");  }  }  } |
| |  | | --- | | A.  A |  |  | | --- | | B.  B |  |  | | --- | | C.  Compilation Error |  |  | | --- | | D.  Run Time Error | | | |
|  | | |

|  |  |
| --- | --- |
| **87.** | What is the output of this program?  Note : Execution command line : $ java exception\_handling  class exception\_handling  {  public static void main(String args[])  {  try  {  int a = args.length;  int b = 10 / a;  System.out.print(a);  }  catch (ArithmeticException e)  {  System.out.println("1");  }  }  } |
| |  | | --- | | A.  0 |  |  | | --- | | B.  1 |  |  | | --- | | C.  Compilation Error |  |  | | --- | | D.  Run Time Error | | | |
|  | | |
| **88.** | What is the output of this program?  class exception\_handling  {  public static void main(String args[])  {  try  {  throw new NullPointerException ("Hello");  }  catch(ArithmeticException e)  {  System.out.print("B");  }  }  } |
| |  | | --- | | A.  A |  |  | | --- | | B.  B |  |  | | --- | | C.  Compilation Error |  |  | | --- | | D.  Run Time Error | | | |
|  | | |

|  |  |
| --- | --- |
| **89.** | What is the output of this program?  class Output  {  public static void main(String args[])  {  try  {  int a = 0;  int b = 5;  int c = b / a;  System.out.print("Hello");  }  catch(Exception e)  {  System.out.print("World");  }  }  } |
| |  | | --- | | A.  Hello |  |  | | --- | | B.  World |  |  | | --- | | C.  HelloWorld |  |  | | --- | | D.  Hello World | | | |
|  | | |
| **90.** | What is the output of this program?  class Output  {  public static void main(String args[])  {  try  {  int a = 0;  int b = 5;  int c = a / b;  System.out.print("Hello");  }  catch(Exception e)  {  System.out.print("World");  }  }  } |
| |  | | --- | | A.  Hello |  |  | | --- | | B.  World |  |  | | --- | | C.  HelloWorld |  |  | | --- | | D.  Compilation Error | | | |
|  | | |

|  |  |
| --- | --- |
| **91.** | What is the output of this program?  class Myexception extends Exception  {  int detail;  Myexception(int a)  {  detail = a;  }  public String toString()  {  return "detail";  }  }  class Output  {  static void compute (int a) throws Myexception  {  throw new Myexception(a);  }  public static void main(String args[])  {  try  {  compute(3);  }  catch(Myexception e)  {  System.out.print("Exception");  }  }  } |
| |  | | --- | | A.  3 |  |  | | --- | | B.  Exception |  |  | | --- | | C.  Runtime Error |  |  | | --- | | D.  Compilation Error | | | |
|  | | |
| **92.** | What is the output of this program?  class Myexception extends Exception  {  int detail;  Myexception(int a)  {  detail = a;  }  public String toString()  {  return "detail";  }  }  class Output  {  static void compute (int a) throws Myexception  {  throw new Myexception(a);  }  public static void main(String args[])  {  try  {  compute(3);  }  catch(DevideByZeroException e)  {  System.out.print("Exception");  }  }  } |
| |  | | --- | | A.  3 |  |  | | --- | | B.  Exception |  |  | | --- | | C.  Runtime Error |  |  | | --- | | D.  Compilation Error | | | |
|  | | |

|  |  |
| --- | --- |
| **93.** | What is the output of this program?  class exception\_handling  {  public static void main(String args[])  {  try  {  System.out.print("Hello" + " " + 1 / 0);  }  finally  {  System.out.print("World");  }  }  } |
| |  | | --- | | A.  Hello |  |  | | --- | | B.  World |  |  | | --- | | C.  Compilation Error |  |  | | --- | | D.  First Exception then World | | | |
|  | | |
| **94.** | What is the output of this program?  class exception\_handling  {  public static void main(String args[])  {  try  {  int i, sum;  sum = 10;  for (i = -1; i < 3 ;++i)  {  sum = (sum / i);  System.out.print(i);  }  }  catch(ArithmeticException e)  {  System.out.print("0");  }  }  } |
| |  | | --- | | A.  -1 |  |  | | --- | | B.  0 |  |  | | --- | | C.  -10 |  |  | | --- | | D.  -101 |  |  | | --- | | E.  None | | | |
|  | | |

|  |  |
| --- | --- |
| **95.** | When does Exceptions in Java arises in code sequence? |
| |  | | --- | | A.  Run Time |  |  | | --- | | B.  compilation fail |  |  | | --- | | C.  Can occur any time |  |  | | --- | | D.  None | | | |
|  | | |
| **96.** | Which of these keywords is not a part of exception handling? |
| |  | | --- | | A.  try |  |  | | --- | | B.  finally |  |  | | --- | | C.  thrown |  |  | | --- | | D.  catch | | | |
|  | | |

|  |  |
| --- | --- |
| **97.** | Which of these keywords must be used to monitor for exceptions? |
| |  | | --- | | A.  try |  |  | | --- | | B.  finally |  |  | | --- | | C.  throw |  |  | | --- | | D.  catch | | | | |
|  | | | |
| **98.** | Which of these keywords must be used to handle the exception thrown by try block in some rational manner? | |
| |  | | --- | | A.  try |  |  | | --- | | B.  finally |  |  | | --- | | C.  throw |  |  | | --- | | D.  catch | | | | |
|  | | | |

|  |  |  |
| --- | --- | --- |
| **99.** | Which of these keywords is used to manually throw an exception? | |
| |  | | --- | | A.  try |  |  | | --- | | B.  finally |  |  | | --- | | C.  throw |  |  | | --- | | D.  catch | | | | |
|  | | | |
| **100.** | | Which of these is a super class of all exceptional type classes? |
| |  | | --- | | A.  String |  |  | | --- | | B.  RunTimeException |  |  | | --- | | C.  Throwable |  |  | | --- | | D.  Cachable | | | | |
|  | | | |

|  |  |  |
| --- | --- | --- |
| **101.** | Which of these class is related to all the exceptions that can be caught by using catch? | |
| |  | | --- | | A.  Error |  |  | | --- | | B.  Exception |  |  | | --- | | C.  RunTimeException |  |  | | --- | | D.  All the above | | | | |
|  | | | |
| **102.** | Which of these class is related to all the exceptions that cannot be caught? |
| |  | | --- | | A.  Error |  |  | | --- | | B.  Exception |  |  | | --- | | C.  RunTimeException |  |  | | --- | | D.  All the above | | | | |
|  | | | |

|  |  |
| --- | --- |
| **103.** | Which of these handles the exception when no catch is used? |
| |  | | --- | | A.  Default handler |  |  | | --- | | B.  finally |  |  | | --- | | C.  throw handler |  |  | | --- | | D.  Java run time system | | | |
|  | | |
| **104.** | Which of these keywords is used to manually throw an exception? |
| |  | | --- | | A.  try |  |  | | --- | | B.  finally |  |  | | --- | | C.  throw |  |  | | --- | | D.  catch | | | |
|  | | |

|  |  |
| --- | --- |
| **105.** | Which of these keywords is used to generate an exception explicitly? |
| |  | | --- | | A.  try |  |  | | --- | | B.  finally |  |  | | --- | | C.  throw |  |  | | --- | | D.  catch | | | |
|  | | |
| **106.** | Which of these class is related to all the exceptions that are explicitly thrown? |
| |  | | --- | | A.  Error |  |  | | --- | | B.  Exception |  |  | | --- | | C.  Throwable |  |  | | --- | | D.  Throw | | | |
|  | | |

|  |  |
| --- | --- |
| **107.** | Which of these operator is used to generate an instance of an exception than can be thrown by using throw? |
| |  | | --- | | A.  new |  |  | | --- | | B.  malloc |  |  | | --- | | C.  alloc |  |  | | --- | | D.  thrown | | | | |
|  | | | |
| **108.** | Which of these keywords is used to by the calling function to guard against the exception that is thrown by called function? | |
| |  | | --- | | A.  try |  |  | | --- | | B.  throw |  |  | | --- | | C.  throws |  |  | | --- | | D.  catch | | | | |
|  | | | |

|  |  |
| --- | --- |
| **109.** | Which part of code gets executed whether exception is caught or not? |
| |  | | --- | | A.  try |  |  | | --- | | B.  finally |  |  | | --- | | C.  catch |  |  | | --- | | D.  throw | | | |
|  | | |
| **110.** | Which of the following should be true of the object thrown by a thrown statement? |
| |  | | --- | | A.  Should be assignable to String type |  |  | | --- | | B.  Should be assignable to Exception type |  |  | | --- | | C.  Should be assignable to Throwable type |  |  | | --- | | D.  Should be assignable to Error type | | | |
|  | | |

|  |  |
| --- | --- |
| **111.** | At runtime, error is recoverable. |
| |  | | --- | | A.  True |  |  | | --- | | B.  False |  |  | | --- | | C.  May be |  |  | | --- | | D.  Can't say | | | |
|  | | |
| **112.** | Which of these clause will be executed even if no exceptions are found? |
| |  | | --- | | A.  finally |  |  | | --- | | B.  throws |  |  | | --- | | C.  throw |  |  | | --- | | D.  catch | | | |
|  | | |

|  |  |
| --- | --- |
| **113.** | Which of these keywords are used for the block to be examined for exceptions? |
| |  | | --- | | A.  try |  |  | | --- | | B.  catch |  |  | | --- | | C.  throw |  |  | | --- | | D.  check | | | |
|  | | |
| **114.** | Which of these statements is incorrect? |
| |  | | --- | | A.  try block need not to be followed by catch block |  |  | | --- | | B.  try block can be followed by finally block instead of catch block |  |  | | --- | | C.  try can be followed by both catch and finally block |  |  | | --- | | D.  try need not to be followed by anything | | | |
|  | | |

|  |  |
| --- | --- |
| **115.** | Which of these classes is used to define exceptions? |
| |  | | --- | | A.  Exception |  |  | | --- | | B.  Throwable |  |  | | --- | | C.  Abstract |  |  | | --- | | D.  System | | | |
|  | | |
| **116.** | Which of these methods return description of an exception? |
| |  | | --- | | A.  getException() |  |  | | --- | | B.  getMessage() |  |  | | --- | | C.  obtainDescription() |  |  | | --- | | D.  obtainException() | | | |
|  | | |

|  |  |
| --- | --- |
| **117.** | Which of these methods is used to print stack trace? |
| |  | | --- | | A.  obtainStackTrace() |  |  | | --- | | B.  printStackTrace() |  |  | | --- | | C.  getStackTrace() |  |  | | --- | | D.  displayStackTrace() | | | |
|  | | |
| **118.** | Which of these methods return localized description of an exception? |
| |  | | --- | | A.  getLocalizedMessage() |  |  | | --- | | B.  getMessage() |  |  | | --- | | C.  obtainLocalizedMessage() |  |  | | --- | | D.  printLocalizedMessage() | | | |
|  | | |

|  |  |
| --- | --- |
| **119.** | Which of these classes is super class of Exception class? |
| |  | | --- | | A.  Throwable |  |  | | --- | | B.  System |  |  | | --- | | C.  RunTime |  |  | | --- | | D.  class | | | |
|  | | |
| **120.** | Which of these class is related to all the exceptions that cannot be caught? |
| |  | | --- | | A.  Error |  |  | | --- | | B.  Exception |  |  | | --- | | C.  RunTimeException |  |  | | --- | | D.  All the above | | | |
|  | | |

|  |  |
| --- | --- |
| **121.** | What exception thrown by parseInt() method? |
| |  | | --- | | A.  AirthmeticException |  |  | | --- | | B.  ClassNotFoundException |  |  | | --- | | C.  NullPinterException |  |  | | --- | | D.  NumberFormatException | | | |
|  | | |
| **122.** | What is the output of this program?  public class Bits  {  public static void main(String[] args)  {  try  {  return;  }  finally  {  System.out.println( "Finally" );  }  }  } |
| |  | | --- | | A.  finally |  |  | | --- | | B.  compilation fail |  |  | | --- | | C.  the code runn with no output |  |  | | --- | | D.  An exception is thrown at runtime. | | | |
|  | | |

|  |  |
| --- | --- |
| **123.** | What is the output of this program?  class exception\_handling  {  public static void main(String args[])  {  try  {  int a = 1;  int b = 10 / a;  try  {  if (a == 1)  a = a / a - a;  if (a == 2)  {  int c[] = {1};  c[8] = 9;  }  }  finally  {  System.out.print("A");  }  }  catch (Exception e)  {  System.out.println("B");  }  }  } |
| |  | | --- | | A.  A |  |  | | --- | | B.  B |  |  | | --- | | C.  AB |  |  | | --- | | D.  BA |  |  | | --- | | E.  None | | | |
|  | | |
| **124.** | What is the output of this program?  Note: Execution command line: $ java exception\_handling one two  class exception\_handling  {  public static void main(String args[])  {  try  {  int a = args.length;  int b = 10 / a;  System.out.print(a);  try  {  if (a == 1)  a = a / a - a;  if (a == 2)  {  int []c = {1};  c[8] = 9;  }  }  catch (ArrayIndexOutOfBoundException e)  {  System.out.println("TypeA");  }  catch (ArithmeticException e)  {  System.out.println("TypeB");  }  }  } |
| |  | | --- | | A.  TypeA |  |  | | --- | | B.  TypeB |  |  | | --- | | C.  Compilation error |  |  | | --- | | D.  Run Time Error | | | |
|  | | |

|  |  |
| --- | --- |
| **125.** | What is the output of this program?  class exception\_handling  {  public static void main(String args[])  {  try  {  throw new NullPointerException ("Hello");  System.out.print("A");  }  catch(ArithmeticException e)  {  System.out.print("B");  }  }  } |
| |  | | --- | | A.  A |  |  | | --- | | B.  B |  |  | | --- | | C.  Compilation Error |  |  | | --- | | D.  Run Time Error | | | |
|  | | |
| **126.** | Which four can be thrown using the throw statement?  1. Error  2. Event  3. Object  4. Throwable  5. Exception  6. RuntimeException |
| |  | | --- | | A.  1, 2, 3 and 4 |  |  | | --- | | B.  2, 3, 4 and 5 |  |  | | --- | | C.  1, 4, 5 and 6 |  |  | | --- | | D.  2, 4, 5 and 6 | | | |
|  | | |

|  |  |
| --- | --- |
| **127.** | Which statement is true? |
| |  | | --- | | A.  A try statement must have at least one corresponding catch block. |  |  | | --- | | B.  Multiple catch statements can catch the same class of exception more than once. |  |  | | --- | | C.  An Error that might be thrown in a method must be declared as thrown by that method, or be handled within that method. |  |  | | --- | | D.  Except in case of VM shutdown, if a try block starts to execute, a corresponding finally block will always start to execute. | | | |
|  | | |
| **128.** | The class at the top of exception class hierarchy is ................. |
| |  | | --- | | A.  ArithmeticException |  |  | | --- | | B.  Throwable |  |  | | --- | | C.  Object |  |  | | --- | | D.  Exception | | | |
|  | | |

|  |  |
| --- | --- |
| **129.** | In which of the following package Exception class exist? |
| |  | | --- | | A.  java.util |  |  | | --- | | B.  java.file |  |  | | --- | | C.  java.io |  |  | | --- | | D.  java.lang |  |  | | --- | | E.  java.net | | | |
|  | | |
| **130.** | Exception generated in try block is caught in ........... block. |
| |  | | --- | | A.  catch |  |  | | --- | | B.  throw |  |  | | --- | | C.  throws |  |  | | --- | | D.  finally |  |  | | --- | | E.  None | | | |
|  | | |

|  |  |
| --- | --- |
| **131.** | Which exception is thrown when divide by zero statement executes? |
| |  | | --- | | A.  AirthmeticException |  |  | | --- | | B.  NullPointerException |  |  | | --- | | D.  None | | | |
|  | | |
| **132.** | Which keyword is used to specify the exception thrown by method? |
| |  | | --- | | A.  catch |  |  | | --- | | B.  throws |  |  | | --- | | C.  finally |  |  | | --- | | D.  throw | | | |
|  | | |

|  |  |
| --- | --- |
| **133.** | What happen in case of multiple catch blocks? |
| |  | | --- | | A.  Either super or subclass can be caught first. |  |  | | --- | | B.  The superclass exception must be caught first. |  |  | | --- | | C.  The superclass exception cannot caught first. |  |  | | --- | | D.  None | | | |
|  | | |
| **134.** | Which exception is thrown when an array element is accessed beyond the array size? |
| |  | | --- | | A.  ArrayElementOutOfBounds |  |  | | --- | | B.  ArrayIndexOutOfBoundsException |  |  | | --- | | C.  ArrayIndexOutOfBounds |  |  | | --- | | D.  None of these | | | |
|  | | |

|  |  |
| --- | --- |
| **135.** | When the JVM runs out of memory, which exception will be thrown? |
| |  | | --- | | A.  MemoryBoundException |  |  | | --- | | B.  OutOfMemoryError |  |  | | --- | | C.  OutOfRangeException |  |  | | --- | | D.  NullReferanceException | | | |
|  | | |
| **136.** | In Java programming environment, the throw keyword is used |
| |  | | --- | | A.  to generate exception programmatically |  |  | | --- | | B.  to throw exception object |  |  | | --- | | C.  to catch exception object |  |  | | --- | | D.  none of the above | | | |
|  | | |

|  |  |
| --- | --- |
| **137.** | Which exception is thrown by read() method ? |
| |  | | --- | | A.  Exception |  |  | | --- | | B.  FileNotFoundException |  |  | | --- | | C.  ReadException | | | | |
|  | | | |
| **138.** | Attempting to access a character that is outside the bounds of a StringBuffer results in a | |
| |  | | --- | | A.  ArrayIndexOutOfBoundsException |  |  | | --- | | B.  StringOverFlowException |  |  | | --- | | C.  StringException |  |  | | --- | | D.  StringIndexOutOfBoundsException | | | | |
|  | | | |

|  |  |
| --- | --- |
| **139.** | Exception and Error are immediate subclasses of a class called |
| |  | | --- | | A.  Object |  |  | | --- | | B.  Throwable |  |  | | --- | | C.  AWT |  |  | | --- | | D.  Panel | | | |
|  | | |
| **140.** | Which of the following does not deal with exceptions? |
| |  | | --- | | A.  throws |  |  | | --- | | B.  throw |  |  | | --- | | C.  finilize |  |  | | --- | | D.  finally | | | |
|  | | |

|  |  |
| --- | --- |
| **141.** | URL throws an exception called |
| |  | | --- | | A.  IllegalURLException |  |  | | --- | | B.  URLException |  |  | | --- | | C.  MalformedHostException |  |  | | --- | | D.  MalformedURLException | | | |
|  | | |
| **142.** | Which class is base class for all exceptions? |
| |  | | --- | | A.  String |  |  | | --- | | B.  Error |  |  | | --- | | C.  Throwable |  |  | | --- | | D.  RunTimeException | | | |
|  | | |

|  |  |
| --- | --- |
| **143.** | Checked exception caught at |
| |  | | --- | | A.  Compile Time |  |  | | --- | | B.  Run time |  |  | | --- | | C.  Both at compile and runtime |  |  | | --- | | D.  None | | | |
|  | | |
| **144.** | Unchecked exception caught at |
| |  | | --- | | A.  Compile Time |  |  | | --- | | B.  Run time |  |  | | --- | | C.  Both at compile and runtime |  |  | | --- | | D.  None | | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **145.** | What exception can occur in the below java program if we access 5 element in the array that does not exist?  public class TException {  public static void main(String[] args) {  try {  int a[] = { 5, 10, 15, 20 };    System.out.println("Element :" + a[4]);  }  finally{}  }  } | |
| |  | | --- | | A.  ArrayIndexOutOfBoundsException |  |  | | --- | | B.  ArithmeticException |  |  | | --- | | C.  NullPointerException |  |  | | --- | | D.  None | | | | |
|  | | | |
| **146.** | Direct subclass of Throwable in Java |
| |  | | --- | | A.  Exception |  |  | | --- | | B.  Error |  |  | | --- | | C.  Both a and b |  |  | | --- | | D.  None | | | | |
|  | | | |

|  |  |
| --- | --- |
| **147.** | un-checked(runtime) exception in java is/are |
| |  | | --- | | A.  ArrayIndexOutOfBoundsException |  |  | | --- | | B.  AirthmeticException |  |  | | --- | | C.  NullPointerException |  |  | | --- | | D.  All the above | | | |
|  | | |
| **148.** | Incorrect statement(s) about finally block in java exception |
| |  | | --- | | A.  Finally block always follow try catch block |  |  | | --- | | B.  finally block always executes whether exception is handled or not. |  |  | | --- | | C.  There can be multiple finally blocks followed by try catch block. |  |  | | --- | | D.  All are correct | | | |
|  | | |

|  |  |
| --- | --- |
| **149.** | True statement(s) about try catch block |
| |  | | --- | | A.  It is mandatory to have catch block with every try block |  |  | | --- | | B.  There must be only one catch block followed by try block |  |  | | --- | | C.  There can be multiple catch block followed by try block. |  |  | | --- | | D.  All | | | |
|  | | |
| **150.** | True statement about throws and throw is/are |
| |  | | --- | | A.  Throws is used with the method signature. |  |  | | --- | | B.  Throw is used within the method. |  |  | | --- | | C.  using throws multiple exceptions can be declared |  |  | | --- | | D.  All are correct. | | | |
|  | | |

|  |  |
| --- | --- |
| **151.** | Keyword that is not a part of exception handling is |
| |  | | --- | | A.  try |  |  | | --- | | B.  finally |  |  | | --- | | C.  thrown |  |  | | --- | | D.  catch | | | |
|  | | |
| **152.** | can any method override without considering the available throws? |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **153.** | if any new method wants to override a old method with same signature then which exception it needs to throws? | |
| |  | | --- | | A.  checked exception |  |  | | --- | | B.  unchecked exception | | | | |
|  | | | |
| **154.** | if 2 methods overriding each other then both of them needs to throws which exception? |
| |  | | --- | | A.  checked exception |  |  | | --- | | B.  unchecked exception | | | | |
|  | | | |

|  |  |  |
| --- | --- | --- |
| **155.** | if the new method throwing a sub-class exception of the exception which was thorwing by old method,can they override? | |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | | |
|  | | | |
| **156.** | every exception was subclass of which class? |
| |  | | --- | | A.  exception class |  |  | | --- | | B.  abstrct class |  |  | | --- | | C.  throwable class |  |  | | --- | | D.  none of the above | | | | |
|  | | | |

Bottom of Form