|  |  |  |
| --- | --- | --- |
| Sr No | Code | Milli sec |
| 1 | for (long i = 0; i < 1000000; i++)  {  Console.WriteLine("Hello " + i);  } | **3194** |
|  | for (long i = 0; i < 1000000; i += 3)  {  Console.WriteLine("Hello " + i);  Console.WriteLine("Hello " + i);  Console.WriteLine("Hello " + i);  } | **2839** |
| 2 | List<int> list = new LinkedList<int>();  for (int i = 1; i < 1000000; ++i)  {  list.Add(i);  }  int size = list.Count;  for (int j = 0; j < list.Count; ++j)  {  Console.WriteLine("Counter " + j);  } | **1790** |
|  | for (int j = 0; j < size; ++j)  {  Console.WriteLine("Counter " + j);  } | **1704** |
| 3 | for (int i = 0; i < 1000000; ++i)  {  bool isMember = i % 2 == 1;  int discount = isMember ? 10 : 5;  Console.WriteLine("discount :" + discount);  } | **1217** |
|  | for(int i=0 ; i<1000000 ; ++i){  isMember = i%2==1 ?true :false;   discount = isMember ? 10 : 5;   Console.WriteLine("discount :" + discount);  } | **1215** |
| 4 | for (int i = 0; i < 100000; ++i)  {  int day = i % 4;  if (day == 0){  Console.WriteLine("Sunday");  }else if (day == 1){  Console.WriteLine("Monday");  }else if (day == 2){  Console.WriteLine("Tuesday");  }else{  Console.WriteLine("Invalid day");  }  } | **226** |
|  | int day = 4;  for (int i = 0; i < 100000; ++i)  {  day = i % 4;  switch (day)  {  case 1:  Console.WriteLine("Sunday");  break;  case 2:  Console.WriteLine("Monday");  break;  case 3:  Console.WriteLine("Tuesday");  break;  default:  Console.WriteLine("Invalid day");  break;  }  } | **202** |
|  | for (int i = 0; i < 100000; ++i) {  Console.WriteLine(day switch {  1 => "Sunday",  2 => "Monday",  3 => "Tuesday",  \_ => "Invalid day"  });  } | **147** |
| 5 | string result = "";  for (int i = 0; i < 10000; i++)  {  result += i;  Console.WriteLine("result" + result);  } | **1096** |
|  | StringBuilder resultBuilder = new StringBuilder();  for (int i = 0; i < 10000; i++){  resultBuilder.Append(i);  Console.WriteLine(resultBuilder);  } | **1068** |
| 6 | try  {  using (StreamReader reader = new StreamReader("file.txt"))  {  int character;  while ((character = reader.Read()) != -1)  {  Console.Write((char)character);  }  }  }  catch (IOException e)  {  Console.WriteLine(e.StackTrace);  } | **606** |
|  | try  {  using (StreamReader reader = new StreamReader("file.txt")) {  string line;  while ((line = reader.ReadLine()) != null)  {  Console.WriteLine(line);  }  }  }  catch (IOException e){  Console.WriteLine(e.StackTrace);  } | **41** |
| 7. **Use of Logical AND (&&) and OR (||) Operators:** | for (int x = 1; x < 100000; ++x)  {  if (x != 0 && y / x > 10)  {  Console.WriteLine("Without short-circuiting");  }  } | **222** |
|  | for (int i = 1; i < 100000; ++i)  {  if (i != 0 & y / x > 10)  {  Console.WriteLine("With short-circuiting");  }  } | **192** |
| 8 | for (int i = 0; i < 1000000; ++i)  {  if (s.Equals(""))  {  Console.WriteLine("Empty string");  }  } | **2462** |
|  | for (int i = 0; i < 1000000; ++i)  {  if (s.Length == 0)  {  Console.WriteLine("Empty string");  }  } | **1474** |
| 9.Optimize Algorithm Complexity | // Bad: Linear search  for (int index = 0; index < array.Length; index++) {  if (array[index] == target) {  return index;  }  }  // Good: Binary search  Array.Sort(array);  int index = Array.BinarySearch(array, target); |  |
| 10. Fixing the Memory Leak | public class SomeClass {  // Static list declaration  private static List<string> list = new List<string>();  // Method to add item to the list  public void AddToList(string item) {  list.Add(item);  // Ensure the list doesn't grow indefinitely  if (list.Count > 100) {  list.RemoveAt(0);  }  }  } |  |
| 11. **Exceptions** | int numTestLoops = 100000;  public ArrayList TestSystemException()  {  ArrayList al = new ArrayList();  for (int i = 0; i < numTestLoops; i++)  {  object o = null;  if ((i % exceptionFactor) != 0)  {  o = new object();  }  try  {  al.Add(o?.ToString());  }  catch (NullPointerException npe)  {  // Continue to get next string  }  }  // return statement is missing in the original Java code  // Assuming returning ArrayList al  return al;  } |  |
|  | public ArrayList TestCodeException()  {  ArrayList al = new ArrayList();  for (int i = 0; i < numTestLoops; i++)  {  try  {  if ((i % exceptionFactor) == 0)  {  throw new NullReferenceException("Force Exception");  }  object o = new object();  al.Add(o.ToString());  }  catch (NullReferenceException npe)  {  // Continue to get next string  }  }  return al;  } |  |
|  | using System;  using System.Collections;  public ArrayList TestDefensiveProgramming()  {  ArrayList al = new ArrayList();  for (int i = 0; i < numTestLoops; i++)  {  Object o = null;  if ((i % exceptionFactor) != 0)  {  o = new Object();  }  if (o != null)  {  al.Add(o.ToString());  }  }  return al;  } |  |
| **12. Choosing a Parser - XML StAX parser** | XmlStreamReader reader = staxFactory.createXMLStreamReader(ins);  while (reader.hasNext())  {  reader.next();  int state = reader.getEventType();  switch (state)  {  case XMLStreamConstants.START\_ELEMENT:  string elementName = reader.getLocalName();  if (ITEM\_ID.Equals(elementName))  {  isItemID = true;  }  break;  case XMLStreamConstants.CHARACTERS:  if (isItemID)  {  string id = reader.getText();  isItemID = false;  if (addItemId(id))  {  return;  }  }  break;  default:  break;  } //end switch  } |  |
|  | while (parser.HasNext())  {  Event event = parser.Next();  switch (event)  {  case Event.KEY\_NAME:  string s = parser.GetString();  if (ITEM\_ID.Equals(s))  {  isItemID = true;  }  break;  case Event.VALUE\_STRING:  if (isItemID)  {  if (AddItemId(parser.GetString()))  {  return;  }  isItemID = false;  }  continue;  default:  continue;  }  } |  |
| **Push parsers (SAX)** | protected class CustomizedInnerHandler : DefaultHandler  {  public void startElement(string space, string name, string raw, Attributes atts)  {  if (name.Length == 0)  {  name = raw;  }  if (name.Equals(ITEM\_ID, StringComparison.OrdinalIgnoreCase))  {  isItemID = true;  }  }  public void characters(char[] ch, int start, int length)  {  if (isItemID)  {  string s = new string(ch, start, length);  isItemID = false;  if (addItemId(s))  {  throw new SAXDoneException("Done");  }  }  }  } |  |