

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ArraylistTest
{
    public class ArrayList
    {
        public Student getValueAt(int pos)
        {
            return data[pos];
        }
        public void nz()
        {
            int q = getLength(); p=q-1;
            Student[] a = new Student[q];
            for(int i=0; i<q; i++)
            {
                data[i] = a[i];
            }
        }
        public ArrayList dividedByboy()
        {
            ArrayList boy = new ArrayList(this.b);
            int b = 0;
            for (int i = 0; i < this.length; i++)
            {
                if (this.data[i].gender == "man")
                {
                    boy.data[b] = this.data[i]; boy.b++;
                }
            }
            return boy;
        }
        public ArrayList dividedBygirl()
        {
            ArrayList girl = new ArrayList(this.g);
            int b = 0;
            for (int i = 0; i < this.length; i++)
            {
                if (this.data[i].gender == "girl")
                {
                    girl.data[b] = this.data[i]; girl.b++;
                }
            }
            return girl;
        }
        public void getboyandgirl()
        {
            this.b = 0; this.g = 0;
            for (int i = 0; i < this.length; i++)
            {
                if (this.data[i].gender == "girl")
                {
                    this.g++;
                }
                else
                {
                    this.b++;
                }
            }
        }
    }
}

```

Continuously maintaining architecture quality of your IT applications is far more strategic and cost effective than postponing issues for future fixing

Architecture Review

You want to continuously deliver good quality software, therefore keeping technical and architectural debts to the minimum.

Unfortunately many companies are struggling to manage technical debts :

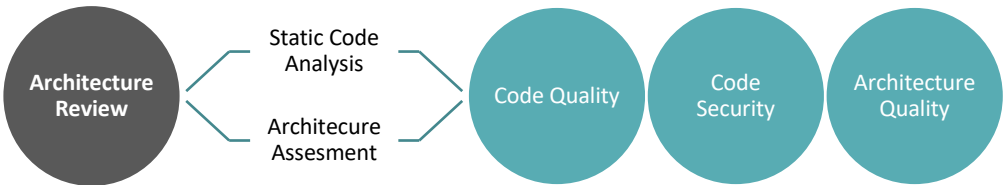
- Absence of governance to track and control technical debts
- Limited capability and capacity to correct and prevent technical debts
- Technical debts make systems unstable and hard to change
- Absence of decision framework to replace or renovate poor quality applications.

Immediate Actions

1. Schedule a 60-mins meeting to tell us your architecture debt story.
2. We will get back with a proposed approach in 1-2 weeks.
3. Conclude technical and commercial discussion.
4. Kick-off project.

By engaging our Architecture Review service you will have clear visibility of technical debts and how to deal with it

We'd love to make the process lean so you can address current issues faster.



Ignoring technical and architectural debts will make your IT applications “decay” and increasingly harder to change – your business will suffer from system instability and long time-to-market

20K+ systems evaluated - 200+ Bn lines of code analyzed - 300+ technologies supported

Contact us at sales@ked-consulting.com. We'd love to help you succeed!

About KED Consulting

KED Consulting is an independent and strategic IT advisory, focusing on helping clients to simplify IT architecture and optimize overall IT performance. Please visit www.ked-consulting.com to learn more. © 2024 PT Karya Ekosistem Digital. All rights reserved