```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Drawing;
//extra
using System.Resources;
using System.Globalization;
using System.Collections;

namespace LOLTeamCounterPick.Classes
{
    class Champion
    {
        #region Private Variables
        private Image _img;
        private int _id;
        private string _name;
        private Image _imgbw;
        #endregion

        #region Public Variables
        public Image img { get { return _img; } }
        public int id { get { return _id; } set { _id = value; } }
        public string name { get { return _name; } }
        public Image imgbw { get { return _imgbw; } }
        #endregion

        #region Private Methods
        private string FixedName(string m_name)
        {
            switch (m_name)
            {
                case "Chogath":
                    return "Cho'Gath";
                case "DrMundo":
                    return "Dr. Mundo";
                case "JarvanIV":
                    return "Jarvan IV";
                case "KhaZix":
                    return "Kha'Zix";
                case "KogMaw":
                    return "Kog'Maw";
                case "LeeSin":
                    return "Lee Sin";
                case "MasterYi":
                    return "Master Yi";
                case "MissFortune":
                    return "Miss Fortune";
                case "TwistedFate":
                    return "Twisted Fate";
                case "VelKoz":
                    return "Vel'Koz";
                case "XinZhao":
                    return "Xin Zhao";
                case "Leblanc":
                    return "LeBlanc";
                default:
                    return m_name;
            }

        }
        private Image ConvertToBlackAndWhite(Image img, int type)
        {
            Bitmap bmp = new Bitmap(img);

            int height = bmp.Height;
            int width = bmp.Width;
            Bitmap newbmp = new Bitmap(width, height);

            LockBitmap lbmp = new LockBitmap(bmp);
            LockBitmap newlbmp = new LockBitmap(newbmp);
            lbmp.LockBits();
```

```
            newlbmp.LockBits();

            Color pixel;
            for (int x = 0; x < width; x++)
            {
                for (int y = 0; y < height; y++)
                {
                    pixel = lbmp.GetPixel(x, y);
                    int r, g, b, Result = 0;
                    r = pixel.R;
                    g = pixel.G;
                    b = pixel.B;
                    switch (type)
                    {
                        case 0://
                            Result = ((r + g + b) / 3);
                            break;
                        case 1://
                            Result = r > g ? r : g;
                            Result = Result > b ? Result : b;
                            break;
                        case 2://
                            Result = ((int)(0.3 * r) + (int)(0.59 * g) + (int)(0.11 * b));
                            break;
                    }
                    newlbmp.SetPixel(x, y, Color.FromArgb(Result, Result, Result));
                }
            }
            lbmp.UnlockBits();
            newlbmp.UnlockBits();


            return (Image)newbmp;
        }
        #endregion

        #region Public Methods
        public Champion(Image m_img, int m_id, string m_name)
        {
            _img = m_img;
            _id = m_id;
            _name = FixedName(m_name);
            _imgbw = ConvertToBlackAndWhite(_img,0);
        }
        #endregion
    }

    class ChampionList
    {
        #region Private Variables
        private List<Champion> _list;
        #endregion

        #region Public Variables
        public List<Champion> list { get { return _list; } }
        #endregion

        #region Private Methods
        #endregion

        #region Public Methods
        public ChampionList() { _list = new List<Champion>(); }
        public ChampionList(ResourceManager rm)
        {
            _list = new List<Champion>();
            ResourceSet resourceSet = rm.GetResourceSet(CultureInfo.CurrentUICulture, true, true);
            foreach (DictionaryEntry entry in resourceSet)
            {
                string tmp = entry.Key.ToString();
                int index = tmp.IndexOf("_Square_0");
                tmp = tmp.Substring(0, index);
                Image img = (Image)entry.Value;
                _list.Add(new Champion(img, 0, tmp));
            }
```

```
            _list = _list.OrderBy(i => i.name).ToList<Champion>();
            for (int i = 0; i < _list.Count; i++)
            {
                _list[i].id = i;
            }
        }
        #endregion
    }

}
```