Practical - 5
Exceptions and Generics

1. Write a program to accept a number from the user and throw an exception if the number is not an odd number.

#code:

```
using System;
namespace Practical_5
{
    class Program
    {
        static void Main(string[] args)
        {
            int num;
            Console.WriteLine("Enter a number:");
            num = Convert.ToInt32(Console.ReadLine());
            try
            {
                if (num % 2 == 0)
                {
                    throw new Exception("Number is not an odd
number");
                }
                else
                {
                    Console.WriteLine("Number is an odd number");
                }
            }
            catch (Exception e)
            {
                Console.WriteLine(e.Message);
            }
            Console.ReadLine();
        }
    }
}
```

Output:

```
Enter a number:
5
Number is an odd number
```

2. Write a program to illustrate usage of try multiple catch with finally clause.

#Code:

```
using System;
namespace Practical_5_2
{
    class Program
    {
        static void Main(string[] args)
        {
            try
            {
                int a = 10, b = 0;
                int c = a / b;
                Console.WriteLine("Result is {0}", c);
            }
            catch (DivideByZeroException e)
            {
                Console.WriteLine("Exception caught: {0}", e);
            }
            catch (Exception e)
            {
                Console.WriteLine("Exception caught: {0}", e);
            }
            finally
            {
                Console.WriteLine("Finally block");
            }
            Console.WriteLine("Rest of the code");
            Console.ReadLine();
        }
    }
}
```

Output:

```
Exception caught: System.DivideByZeroException: Attempted to divide by zero.
   at Practical_5_2.Program.Main(String[] args) in C:\Users\dell\source\repos\Practical-5\Practical-5_2\Program.cs:line 11
Finally block
Rest of the code
```

3. Write a program for creation of user defined exception to show whether candidate is eligible to caste vote.

#Code:

```
using System;
namespace Practical_5_3
{
```

```
class AgeException : Exception
{
   public AgeException(string message) : base(message)
   {
   }
}
class Program
{
   static void Main(string[] args)
   {
      try
      {
         Console.WriteLine("Enter Your Age");
         int age = int.Parse(Console.ReadLine());
         if (age < 18)
         {
            throw new AgeException("You are not eligible to vote");
         }
         else
         {
            Console.WriteLine("You are eligible to vote");
         }
      }
      catch (AgeException e)
      {
         Console.WriteLine(e.Message);
      }
      Console.ReadKey();
   }
}
```

Output:

4. Write a program to calculate area of different shapes using Generic delegate.

```
Enter Your Age
17
You are not eligible to vote
```

#Code:

```
using System;
using System.Collections.Generic;
namespace Practical_5_4
{
```

```
    public delegate void AreaDelegate<T>(params T[] values);
    public class Rectangle
    {
        public void Area(params double[] dimensions)
        {
            Console.WriteLine("Area of Rectangle is : " + (dimensions[0]
* dimensions[1]));
        }
    }
    public class Circle
    {
        public void Area(params double[] dimensions)
        {
            Console.WriteLine("Area of Circle is : " + (3.14 *
dimensions[0] * dimensions[0]));
        }
    }
    public class Triangle
    {
        public void Area(params double[] dimensions)
        {
            Console.WriteLine("Area  of  Triangle  is  :  " +  (0.5  *
dimensions[0] * dimensions[1]));
        }
    }
    public class Square
    {
        public void Area(params double[] dimensions)
        {
            Console.WriteLine("Area of Square is : " + (dimensions[0] *
dimensions[0]));
        }
    }
    class Program
    {
        static void Main(string[] args)
        {
            Rectangle r = new Rectangle();
            Circle c = new Circle();
            Triangle t = new Triangle();
            Square s = new Square();
            AreaDelegate<double> ad = new AreaDelegate<double>(r.Area);
            ad += r.Area;
            ad += c.Area;
            ad += t.Area;
            ad += s.Area;
```

```
            ad.Invoke(10, 20);
            ad.Invoke(10);
            ad.Invoke(10, 20);
            ad.Invoke(10);
            Console.ReadLine();
        }
    }
}
```

Output:

5. Write a program to search color in given ArrayList of colors.

```
Area of Rectangle is : 200
Area of Circle is : 314
Area of Triangle is : 100
Area of Square is : 100
```

#Simple ArrayList:

```
using System;
using System.Collections;
namespace Practical_5_5
{
    class Program
    {
        static void Main(string[] args)
        {
            ArrayList arrayList = new ArrayList();
            arrayList.Add("red");
            arrayList.Add("blue");
            arrayList.Add("green");
            arrayList.Add("yellow");
            arrayList.Add("pink");
            arrayList.Add("black");
            arrayList.Add("white");
            arrayList.Add("orange");
            arrayList.Add("purple");
            arrayList.Add("brown");
            Console.WriteLine("Enter color to search");
            string color = Console.ReadLine();
            if (arrayList.Contains(color))
            {
                Console.WriteLine("Color found");
            }
            else
            {
                Console.WriteLine("Color not found");
```

```
            }
            Console.ReadLine();
        }
    }
}
```

Output:

#Generic List:

```
Enter color to search
purple
Color found
```

```
using System;
using System.Collections.Generic;
namespace Practical_5_5
{
    class Program
    {
        static void Main(string[] args)
        {
            List<string> arrayList = new List<string>();
            arrayList.Add("red");
            arrayList.Add("blue");
            arrayList.Add("green");
            arrayList.Add("yellow");
            arrayList.Add("pink");
            arrayList.Add("black");
            arrayList.Add("white");
            arrayList.Add("orange");
            arrayList.Add("purple");
            arrayList.Add("brown");
            Console.WriteLine("Enter color to search");
            string color = Console.ReadLine();
            if (arrayList.Contains(color))
            {
                Console.WriteLine("Color found");
            }
            else
            {
                Console.WriteLine("Color not found");
            }
            Console.ReadLine();
        }
    }
}
```

Output:

```
Enter color to search
dark green
Color not found
```

6. Write a program to create a generic queue/stack and perform insert, delete and display operations on it.

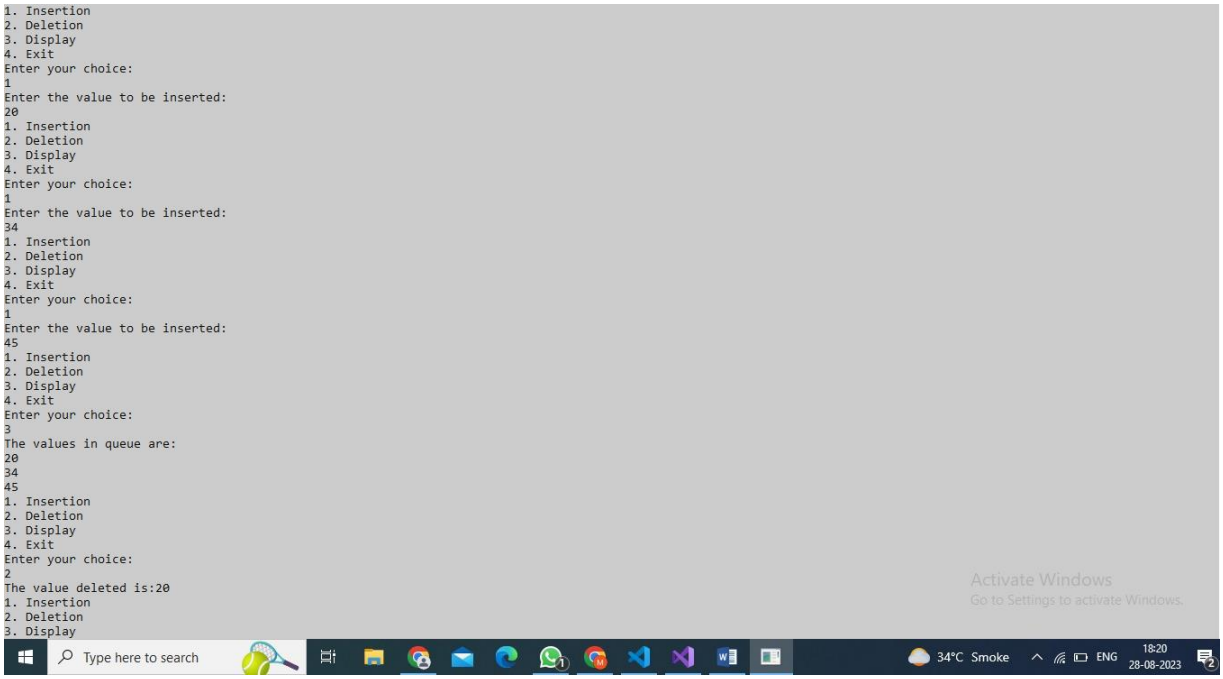
#Code:

```
using System;
using System.Collections.Generic;

namespace Practical_5_6
{
    class Program
    {
        static void Main(string[] args)
        {
            int choice, val;
            Queue<int> q = new Queue<int>();
            while (true)
            {
                Console.WriteLine("1. Insertion");
                Console.WriteLine("2. Deletion");
                Console.WriteLine("3. Display");
                Console.WriteLine("4. Exit");
                Console.WriteLine("Enter your choice:");
                choice = Convert.ToInt32(Console.ReadLine());
                switch (choice)
                {
                    case 1:
                        Console.WriteLine("Enter the value to be
inserted:");
                        val = Convert.ToInt32(Console.ReadLine());
                        q.Enqueue(val);
                        break;
                    case 2:
                        Console.WriteLine("The value deleted is:" +
q.Dequeue());
                        break;
                    case 3:
                        Console.WriteLine("The values in queue are:");
                        foreach (int i in q)
                            Console.WriteLine(i);
                        break;
```

```
                    case 4:
                        Environment.Exit(0);
                        break;
                }
            }
        }
    }
}
```

Output:



7. Write a program to create a nongeneric queue/stack and perform insert, delete and display operations on it.

```
using System;
using System.Collections;
namespace Practical_5_7
{
    class Program
    {
        static void Main(string[] args)
        {
            Stack st = new Stack();
            st.Push("Hello");
            st.Push("World");
            st.Push("!");
            Console.WriteLine("Current stack: ");
            foreach (string s in st)
            {
                Console.Write(s + " ");
```

```
            }
            Console.WriteLine();
            st.Push("Goodbye");
            Console.WriteLine("The next poppable value in stack: {0}",
st.Peek());
            Console.WriteLine("Current stack: ");
            foreach (string s in st)
            {
                Console.Write(s + " ");
            }
            Console.WriteLine();
            Console.WriteLine("Removing values ");
            st.Pop();
            st.Pop();
            st.Pop();
            Console.WriteLine("Current stack: ");
            foreach (string s in st)
            {
                Console.Write(s + " ");
            }
            Console.ReadKey();
        }
    }
}
```

Output:

```
Current stack:
! World Hello
The next poppable value in stack: Goodbye
Current stack:
Goodbye ! World Hello
Removing values
Current stack:
Hello
```