Spring Framework Exercise

(1) Write a program to demonstrate Tightly Coupled code.

```
package com.framework.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
@SpringBootApplication
public class SpringFrameworkday1ExerciseApplication {
    public static void main(String[] args) {
        BinarySearchImpl bs = new BinarySearchImpl();
        int result = bs.binarySearch(new int[] {10, 6, 2, 9}, 6);
        if (result == -1)
            System.out.println("Element not present");
        else
            System.out.println("Element found at index: " + result);
        //SpringApplication.run(SpringFrameworkday1ExerciseApplication.class,
args);
    }
}
```

```
package com.framework.demo;

public class BinarySearchImpl {
    public int binarySearch(int[] numbers, int numberToSearchFor) {
        BubbleSortAlgorithm bubbleSortAlgorithm = new
BubbleSortAlgorithm();
        int[] sortedNumbers = bubbleSortAlgorithm.sort(numbers);
        System.out.println("Sorted Array List");
        for(int element: sortedNumbers)
            System.out.println(element);
        // Search the array
        int n = sortedNumbers.length;
        int res = binarySearch(sortedNumbers, 0, n - 1, numberToSearchFor);
        if (res >= 0)
            return res;
        else
            return -1;
    }
    public int binarySearch(int arr[], int l, int r, int x)
    {
        if (r >= l) {
            int mid = l + (r - l) / 2;
            if (arr[mid] == x)
                return mid;
            if (arr[mid] > x)
                return binarySearch(arr, l, mid - 1, x);
```

```
                return binarySearch(arr, mid + 1, r, x);
            }
            return -1;
        }
}
```

```
package com.framework.demo;

public class BubbleSortAlgorithm {
        public int[] sort(int[] arr) {
            // Bubble Sort
            int n = arr.length;
            for (int i = 0; i < n-1; i++)
                for (int j = 0; j < n-i-1; j++)
                    if (arr[j] > arr[j+1])
                    {
                        int temp = arr[j];
                        arr[j] = arr[j+1];
                        arr[j+1] = temp;
                    }
            return arr;
        }
}
```

(2) Write a program to demonstrate Loosely Coupled code.

```
package com.framework.demo;

public interface SortAlgorithm {
        public int[] sort(int[] numbers);
}
```

```
package com.framework.demo;

public class InsertionSortAlgorithm implements SortAlgorithm{
        public int[] sort(int[] arr) {
            int i, key, j;
            int n = arr.length;
            for (i = 1; i < n; i++)
            {
                key = arr[i];
                j = i - 1;
                while (j >= 0 && arr[j] > key)
                {
                    arr[j + 1] = arr[j];
```

```
                j = j - 1;
            }
            arr[j + 1] = key;
        }
        return arr;
    }
}
```

```
package com.framework.demo;

public class BubbleSortAlgorithm implements SortAlgorithm {
    public int[] sort(int[] arr) {
        // Bubble Sort
        int n = arr.length;
        for (int i = 0; i < n-1; i++)
            for (int j = 0; j < n-i-1; j++)
                if (arr[j] > arr[j+1])
                {
                    int temp = arr[j];
                    arr[j] = arr[j+1];
                    arr[j+1] = temp;
                }
        return arr;
    }
}
```

```
package com.framework.demo;

public class BinarySearchImpl {
    private SortAlgorithm sortAlgorithm;
    public BinarySearchImpl(SortAlgorithm sortAlgorithm)
    {
        this.sortAlgorithm = sortAlgorithm;
    }
    public int binarySearch(int[] numbers, int numberToSearchFor) {
        /*
        BubbleSortAlgorithm bubbleSortAlgorithm = new
BubbleSortAlgorithm();
        int[] sortedNumbers = bubbleSortAlgorithm.sort(numbers);
          */
        int[] sortedNumbers = sortAlgorithm.sort(numbers);
        System.out.println("Sorted Array List");
        for(int element: sortedNumbers)
            System.out.println(element);
        // Search the array
        int n = sortedNumbers.length;
        int res = binarySearch(sortedNumbers, 0, n - 1, numberToSearchFor);
```

```
            if (res >= 0)
                  return res;
            else
                  return -1;
      }
      public int binarySearch(int arr[], int l, int r, int x)
      {
            if (r >= l) {
                  int mid = l + (r - l) / 2;
                  if (arr[mid] == x)
                        return mid;
                  if (arr[mid] > x)
                        return binarySearch(arr, l, mid - 1, x);
                  return binarySearch(arr, mid + 1, r, x);
            }
            return -1;
      }
}
```

```
package com.framework.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
@SpringBootApplication
public class SpringFrameworkday1ExerciseApplication {
    public static void main(String[] args) {
        //BinarySearchImpl bs = new BinarySearchImpl();
        //BinarySearchImpl bs = new BinarySearchImpl(new
BubbleSortAlgorithm());
        BinarySearchImpl bs = new BinarySearchImpl(new
InsertionSortAlgorithm());
        int result = bs.binarySearch(new int[] {10, 6, 2, 9}, 6);
        if (result == -1)
            System.out.println("Element not present");
        else
            System.out.println("Element found at index: " + result);
        //SpringApplication.run(SpringFrameworkday1ExerciseApplication.class,
args);
    }
}
```

```
/usr/lib/jvm/jdk-15.0.2/bin/java ...
Sorted Array List
2
6
9
10
Element found at index: 1

Process finished with exit code 0
```

(3) Use @Compenent and @Autowired annotations to in Loosely Coupled code for dependency management

```
package com.framework.demo;

public interface SortAlgorithm {
	public int[] sort(int[] numbers);
}
```

```
package com.framework.demo;

import org.springframework.stereotype.Component;
@Component
public class BubbleSortAlgorithm implements SortAlgorithm {
	public int[] sort(int[] arr) {
		// Bubble Sort
		System.out.println("Bubble sort Use:");
		int n = arr.length;
		for (int i = 0; i < n-1; i++)
			for (int j = 0; j < n-i-1; j++)
				if (arr[j] > arr[j+1])
				{
					int temp = arr[j];
					arr[j] = arr[j+1];
					arr[j+1] = temp;
				}
		return arr;
	}
}
```

```
package com.framework.demo;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;
@Component
public class BinarySearchImpl {
	@Autowired
	private SortAlgorithm sortAlgorithm;
	public BinarySearchImpl(SortAlgorithm sortAlgorithm)
	{
		this.sortAlgorithm = sortAlgorithm;
	}
	public int binarySearch(int[] numbers, int numberToSearchFor) {
		/*
```

```
		BubbleSortAlgorithm bubbleSortAlgorithm = new
BubbleSortAlgorithm();
		int[] sortedNumbers = bubbleSortAlgorithm.sort(numbers);
		  */
		int[] sortedNumbers = sortAlgorithm.sort(numbers);
		System.out.println("Sorted Array List");
		for(int element: sortedNumbers)
			System.out.println(element);
		// Search the array
		int n = sortedNumbers.length;
		int res = binarySearch(sortedNumbers, 0, n - 1, numberToSearchFor);
		if (res >= 0)
			return res;
		else
			return -1;
	}
	public int binarySearch(int arr[], int l, int r, int x)
	{
		if (r >= l) {
			int mid = l + (r - l) / 2;
			if (arr[mid] == x)
				return mid;
			if (arr[mid] > x)
				return binarySearch(arr, l, mid - 1, x);
			return binarySearch(arr, mid + 1, r, x);
		}
		return -1;
	}
}



package com.framework.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.ApplicationContext;
@SpringBootApplication
public class SpringFrameworkday1ExerciseApplication {
    public static void main(String[] args) {
        //BinarySearchImpl bs = new BinarySearchImpl();
        //BinarySearchImpl bs = new BinarySearchImpl(new
BubbleSortAlgorithm());
        //BinarySearchImpl bs = new BinarySearchImpl(new
InsertionSortAlgorithm());
        ApplicationContext applicationContext =

SpringApplication.run(SpringFrameworkday1ExerciseApplication.class, args);
        BinarySearchImpl bs =
applicationContext.getBean(BinarySearchImpl.class);
        int result = bs.binarySearch(new int[] {10, 6, 2, 9}, 6);
```

```
        if (result == -1)
            System.out.println("Element not present");
        else
            System.out.println("Element found at index: " + result);
    }
}
```

(4) Get a Spring Bean from application context and display its properties.

```
    public static void main(String[] args) {

        //BinarySearchImpl bs = new BinarySearchImpl();

        //BinarySearchImpl bs = new BinarySearchImpl(new BubbleSortAlgorithm());
        //BinarySearchImpl bs = new BinarySearchImpl(new InsertionSortAlgorithm());

        ApplicationContext applicationContext =
                SpringApplication.run(SpringFrameworkday1ExerciseApplication.class, args);
        BinarySearchImpl bs = applicationContext.getBean(BinarySearchImpl.class);

        System.out.println("Property of Bean : ");
        System.out.println("Application Name : "+applicationContext.getApplicationName());

        System.out.println("Display Name : "+applicationContext.getDisplayName());

        System.out.println("Defination Names : "+applicationContext.getBeanDefinitionNames());

        int result = bs.binarySearch(new int[] {10, 6, 2, 9}, numberToSearchFor: 6);
        if (result == -1)
            System.out.println("Element not present");
        else
```

```
Property of Bean :
Application Name :
Display Name : org.springframework.context.annotation.AnnotationConfigApplicationContext@10a41595
Defination Names : [Ljava.lang.String;@3064d5d6
Insertion Sort use
Sorted Array List
2
6
9
10
Element found at index: 1
2021-03-03 20:28:21.746 DEBUG 5973 --- [extShutdownHook] s.c.a.AnnotationConfigApplicationContext : (
```

(5) Demonstrate how you will resolve ambiguity while autowiring bean (Hint : @Primary)

```
2021-03-03 20:33:04.003 ERROR 6320 --- [  restartedMain] o.s.b.d.LoggingFailureAnalysisReporter   :

***************************
APPLICATION FAILED TO START
***************************

Description:

Parameter 0 of constructor in com.framework.demo.BinarySearchImpl required a single bean, but 2 were found:
    - bubbleSortAlgorithm: defined in file [/home/ttn/Documents/Spring_boot_assignments/Spring-framework/target/classes/com/framework/
    - insertionSortAlgorithm: defined in file [/home/ttn/Documents/Spring_boot_assignments/Spring-framework/target/classes/com/framewo


Action:

Consider marking one of the beans as @Primary, updating the consumer to accept multiple beans, or using @Qualifier to identify the bea


Process finished with exit code 0
```

```
package com.framework.demo;

public interface SortAlgorithm {
	public int[] sort(int[] numbers);
}
```

```
package com.framework.demo;

import org.springframework.context.annotation.Primary;
import org.springframework.stereotype.Component;
@Component
@Primary
public class InsertionSortAlgorithm implements SortAlgorithm{
	public int[] sort(int[] arr) {
		System.out.println("Insertion Sort use");
		int i, key, j;
		int n = arr.length;
		for (i = 1; i < n; i++)
		{
			key = arr[i];
			j = i - 1;
			while (j >= 0 && arr[j] > key)
			{
```

```
                arr[j + 1] = arr[j];
                j = j - 1;
            }
            arr[j + 1] = key;
        }
        return arr;
    }
}
```

```
package com.framework.demo;

import org.springframework.stereotype.Component;
@Component
public class BubbleSortAlgorithm implements SortAlgorithm {
    public int[] sort(int[] arr) {
        // Bubble Sort
        System.out.println("Bubble sort Use:");
        int n = arr.length;
        for (int i = 0; i < n-1; i++)
            for (int j = 0; j < n-i-1; j++)
                if (arr[j] > arr[j+1])
                {
                    int temp = arr[j];
                    arr[j] = arr[j+1];
                    arr[j+1] = temp;
                }
        return arr;
    }
}
```

```
package com.framework.demo;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;
@Component
public class BinarySearchImpl {
    @Autowired
    private SortAlgorithm sortAlgorithm;
    public BinarySearchImpl(SortAlgorithm sortAlgorithm)
    {
        this.sortAlgorithm = sortAlgorithm;
    }
    public int binarySearch(int[] numbers, int numberToSearchFor) {
        /*
        BubbleSortAlgorithm bubbleSortAlgorithm = new
BubbleSortAlgorithm();
        int[] sortedNumbers = bubbleSortAlgorithm.sort(numbers);
          */
        int[] sortedNumbers = sortAlgorithm.sort(numbers);
```

```
		System.out.println("Sorted Array List");
		for(int element: sortedNumbers)
			System.out.println(element);
		// Search the array
		int n = sortedNumbers.length;
		int res = binarySearch(sortedNumbers, 0, n - 1, numberToSearchFor);
		if (res >= 0)
			return res;
		else
			return -1;
	}
	public int binarySearch(int arr[], int l, int r, int x)
	{
		if (r >= l) {
			int mid = l + (r - l) / 2;
			if (arr[mid] == x)
				return mid;
			if (arr[mid] > x)
				return binarySearch(arr, l, mid - 1, x);
			return binarySearch(arr, mid + 1, r, x);
		}
		return -1;
	}
}

package com.framework.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.ApplicationContext;
@SpringBootApplication
public class SpringFrameworkday1ExerciseApplication {
    public static void main(String[] args) {
        //BinarySearchImpl bs = new BinarySearchImpl();
        //BinarySearchImpl bs = new BinarySearchImpl(new
BubbleSortAlgorithm());
        //BinarySearchImpl bs = new BinarySearchImpl(new
InsertionSortAlgorithm());
        ApplicationContext applicationContext =

SpringApplication.run(SpringFrameworkday1ExerciseApplication.class, args);
        BinarySearchImpl bs =
applicationContext.getBean(BinarySearchImpl.class);
        int result = bs.binarySearch(new int[] {10, 6, 2, 9}, 6);
        if (result == -1)
            System.out.println("Element not present");
        else
            System.out.println("Element found at index: " + result);
    }
}
```

```
2021-03-03 20:18:50.928 DEBUG 5441 --- [  restartedMain] o.s.boot.devtools.restart.Restar
2021-03-03 20:18:50.928 DEBUG 5441 --- [  restartedMain] o.s.boot.devtools.restart.Restar
2021-03-03 20:18:50.928 DEBUG 5441 --- [  restartedMain] o.s.boot.devtools.restart.Restar
Insertion Sort use
Sorted Array List
2
6
9
10
Element found at index: 1
2021-03-03 20:18:50.944 DEBUG 5441 --- [extShutdownHook] s.c.a.AnnotationConfigApplicatio

Process finished with exit code 0
```

(6) Perform Constructor Injection in a Spring Bean

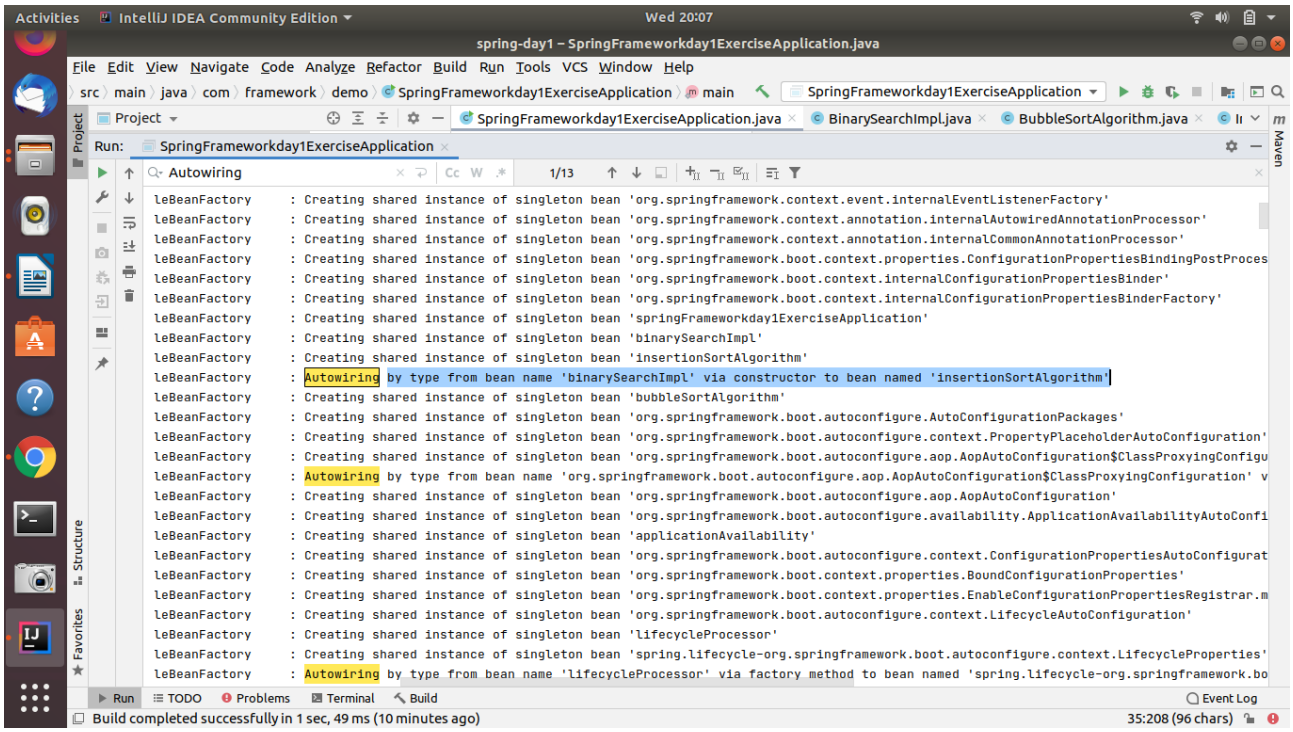
application.properties File

logging.level.org.springframework = debug

```java
@Component
public class BinarySearchImpl {

    @Autowired
    private SortAlgorithm sortAlgorithm;

    //Constructor Injection
    public BinarySearchImpl(SortAlgorithm sortAlgorithm)
    {
        this.sortAlgorithm = sortAlgorithm;
    }
```


Activities
IntelliJ IDEA Community Edition
Wed 20:07
spring-day1 – SpringFrameworkday1ExerciseApplication.java
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
src main java com framework demo SpringFrameworkday1ExerciseApplication main
SpringFrameworkday1ExerciseApplication
Project
SpringFrameworkday1ExerciseApplication.java
BinarySearchImpl.java
BubbleSortAlgorithm.java
Run:
SpringFrameworkday1ExerciseApplication
Autowiring
1/13
leBeanFactory : Creating shared instance of singleton bean 'org.springframework.context.event.internalEventListenerFactory'
leBeanFactory : Creating shared instance of singleton bean 'org.springframework.context.annotation.internalAutowiredAnnotationProcessor'
leBeanFactory : Creating shared instance of singleton bean 'org.springframework.context.annotation.internalCommonAnnotationProcessor'
leBeanFactory : Creating shared instance of singleton bean 'org.springframework.boot.context.properties.ConfigurationPropertiesBindingPostProces
leBeanFactory : Creating shared instance of singleton bean 'org.springframework.boot.context.internalConfigurationPropertiesBinder'
leBeanFactory : Creating shared instance of singleton bean 'org.springframework.boot.context.internalConfigurationPropertiesBinderFactory'
leBeanFactory : Creating shared instance of singleton bean 'springFrameworkday1ExerciseApplication'
leBeanFactory : Creating shared instance of singleton bean 'binarySearchImpl'
leBeanFactory : Creating shared instance of singleton bean 'insertionSortAlgorithm'
leBeanFactory : Autowiring by type from bean name 'binarySearchImpl' via constructor to bean named 'insertionSortAlgorithm'
leBeanFactory : Creating shared instance of singleton bean 'bubbleSortAlgorithm'
leBeanFactory : Creating shared instance of singleton bean 'org.springframework.boot.autoconfigure.AutoConfigurationPackages'
leBeanFactory : Creating shared instance of singleton bean 'org.springframework.boot.autoconfigure.context.PropertyPlaceholderAutoConfiguration'
leBeanFactory : Creating shared instance of singleton bean 'org.springframework.boot.autoconfigure.aop.AopAutoConfiguration$ClassProxyingConfigu
leBeanFactory : Autowiring by type from bean name 'org.springframework.boot.autoconfigure.aop.AopAutoConfiguration$ClassProxyingConfiguration' v
leBeanFactory : Creating shared instance of singleton bean 'org.springframework.boot.autoconfigure.aop.AopAutoConfiguration'
leBeanFactory : Creating shared instance of singleton bean 'org.springframework.boot.autoconfigure.availability.ApplicationAvailabilityAutoConfi
leBeanFactory : Creating shared instance of singleton bean 'applicationAvailability'
leBeanFactory : Creating shared instance of singleton bean 'org.springframework.boot.autoconfigure.context.ConfigurationPropertiesAutoConfigurat
leBeanFactory : Creating shared instance of singleton bean 'org.springframework.boot.context.properties.BoundConfigurationProperties'
leBeanFactory : Creating shared instance of singleton bean 'org.springframework.boot.context.properties.EnableConfigurationPropertiesRegistrar.m
leBeanFactory : Creating shared instance of singleton bean 'org.springframework.boot.autoconfigure.context.LifecycleAutoConfiguration'
leBeanFactory : Creating shared instance of singleton bean 'lifecycleProcessor'
leBeanFactory : Creating shared instance of singleton bean 'spring.lifecycle-org.springframework.boot.autoconfigure.context.LifecycleProperties'
leBeanFactory : Autowiring by type from bean name 'lifecycleProcessor' via factory method to bean named 'spring.lifecycle-org.springframework.bo
Run
TODO
Problems
Terminal
Build
Event Log
Build completed successfully in 1 sec, 49 ms (10 minutes ago)
35:208 (96 chars)
Project
Structure
Favorites
Maven