**Muhammad Rizwan Khalid**

**180459**

**BSCS 6-A**

**Lab no: 02**

**Activity One:**

package javaapplication1;

public class JavaApplication1 {

public static void main(String[] args) {

Time t0 = new Time(); // Set Time to 00:00:00

Time t1 = new Time(11); // Set Time to 11:00:00

Time t2 = new Time(12, 40); // Set Time to 12:40:00

Time t3 = new Time(23, 40, 55); // Set Time to 23:40:55

// Time t4 = new Time(23, 40, 65); // Set Time to 23:40:65

// Print All Times in Universal Format

System.out.println(t0.toUniversalString());

System.out.println(t1.toUniversalString());

System.out.println(t2.toUniversalString());

System.out.println(t3.toUniversalString());

// Print All Times in Standard Format

System.out.println(t0);

System.out.println(t1);

System.out.println(t2);

System.out.println(t3);

}

}

class Time

{

private int hour; // 0 - 23

private int minute; // 0 - 59

private int second; // 0 - 59

Time(){

setTime(0,0,0);

}

Time(int hr){

setTime(hr,0,0);

}

Time(int hr, int min){

setTime(hr,min,0);

}

Time(int hr, int min, int sec){

setTime(hr,min,sec);

}

// set a new time value using universal time; throw an

// exception if the hour, minute or second is invalid

public void setTime( int h, int m, int s )

{

// validate hour, minute and second

if ( ( h >= 0 && h < 24 ) && ( m >= 0 && m < 60 ) && ( s >= 0 && s < 60 ) )

{

hour = h;

minute = m;

second = s;

} // end if

else

throw new IllegalArgumentException("hour, minute and/or second was out of range" );

} // end method setTime

// convert to String in universal-time format (HH:MM:SS)

public String toUniversalString()

{

return String.format( "%02d:%02d:%02d", hour, minute, second );

} // end method toUniversalString

// convert to String in standard-time format (H:MM:SS AM or PM)

public String toString()

{

return String.format( "%d:%02d:%02d %s",

( ( hour == 0 || hour == 12 ) ? 12 : hour % 12 ),

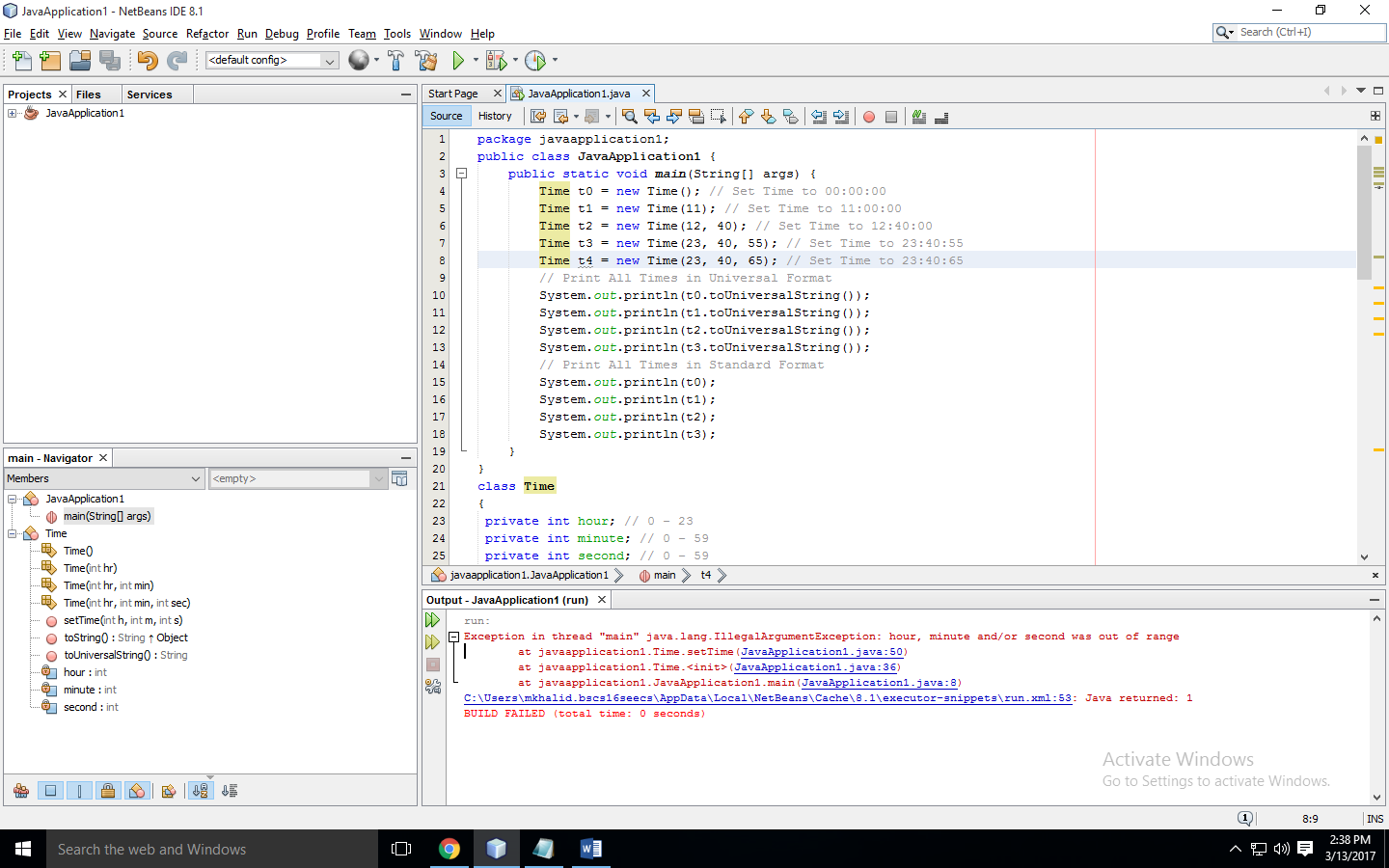
minute, second, ( hour < 12 ? "AM" : "PM" ) );

} // end method toString

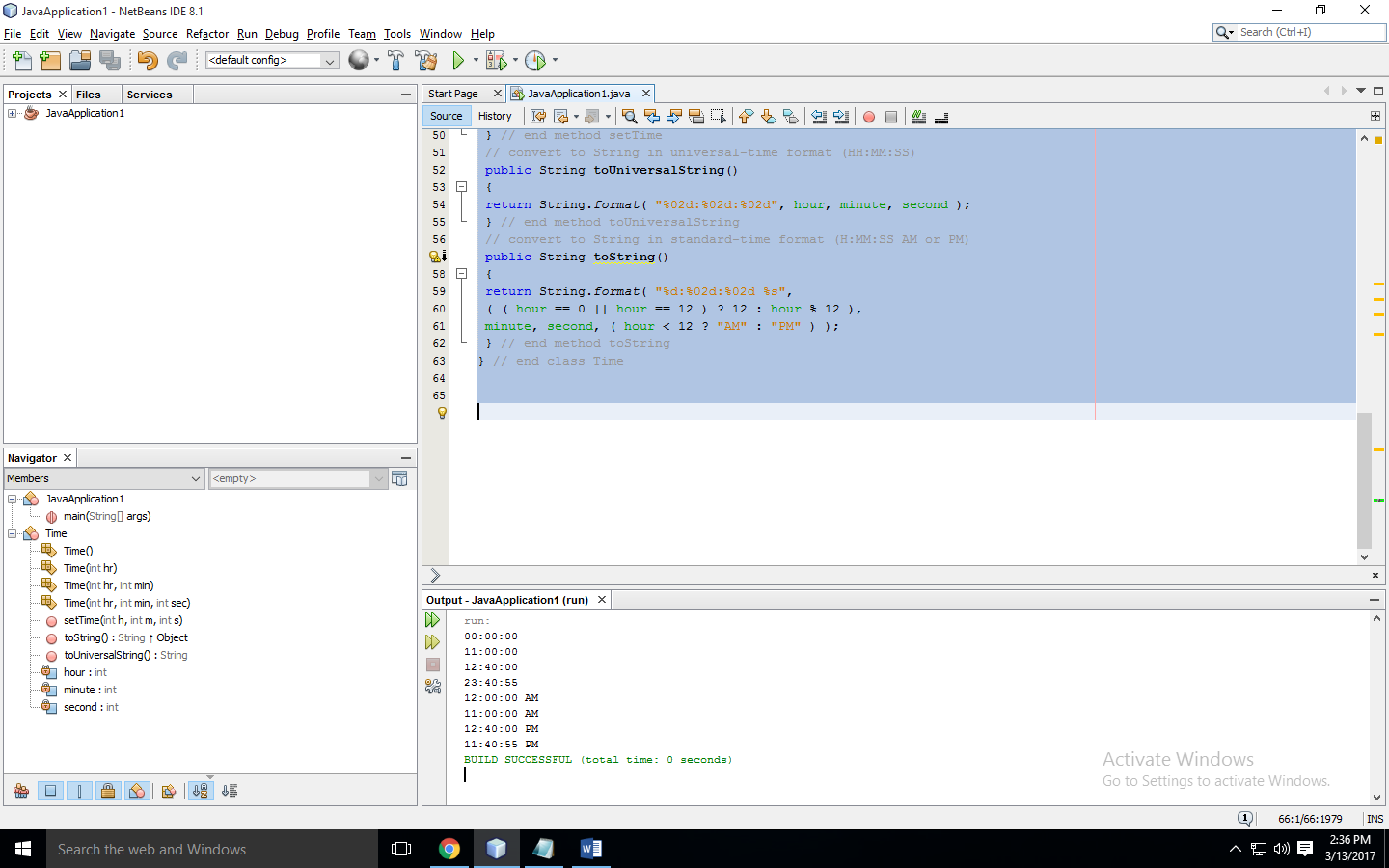
} // end class Time

**Output:**

**With T4:**



**Commenting T4:**



**Activity two:**

package javaapplication1;

public class JavaApplication1 {

public static void main(String[] args) {

Overload ol = new Overload();

double result;

// call all versions of test()

ol.test();

ol.test(10);

ol.test(10, 20);

result = ol.test(123.2);

System.out.println("Result of ol.test(123.2): " + result);

}

}

class Overload {

void test() {

System.out.println("No parameters");

}

// Overload test for one integer parameter.

void test(int a) {

System.out.println("a: " + a);

}

// Overload test for two integer parameters.

void test(int a, int b) {

System.out.println("a and b: " + a + " " + b);

}

// Overload test for a double parameter

double test(double a) {

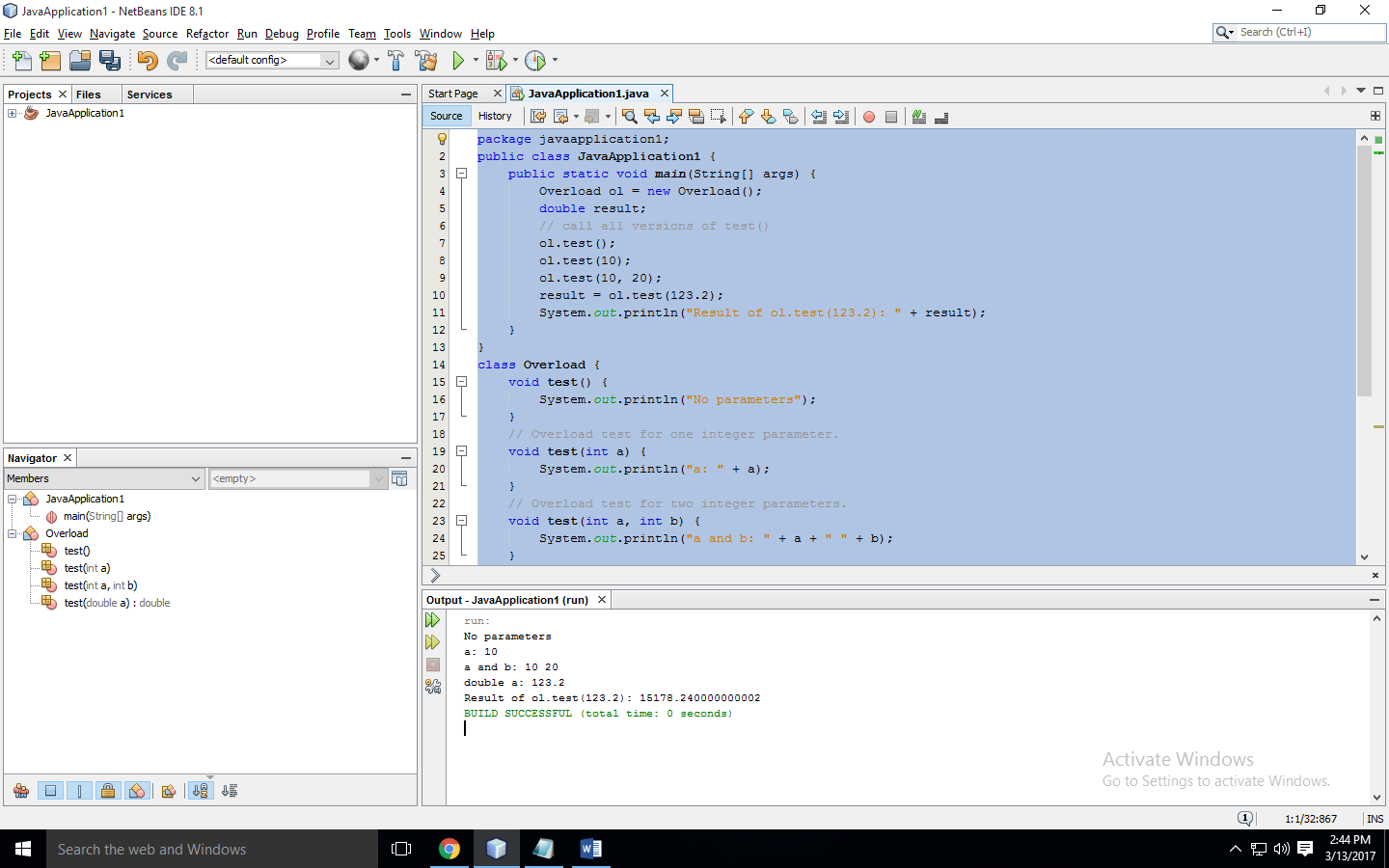
System.out.println("double a: " + a);

return a\*a;

}

}

**Output:**



**Observation:** The concept of overloading is that we can create a single named method having different parameter lists and return type. In this activity there was four methods having single name Test but first method was no argument method. The second method was one argument method. The third method was two argument method while last method was single argument and double return type method. The methods can have different return types but if any of them have same data type list compiler will still give the error.

**Activity Three:**

**Error Observation:** There are two following errors in activity three:

* Firstly, there are two same methods having one argument list although the return type is different but there is the error of method overloading prompted by compiler.
* Secondly, there are two other method having two arguments of same type i.e. int so there arises method overloading error.

**Task one:**

package javaapplication1;

import java.util.Scanner;

public class JavaApplication1 {

public static void main(String[] args) {

int res;

for(int i =1; i <= 1000; i++){

if (isPerfect(i) == true){

res = i;

System.out.print(i+"\t");

for(int j =1;j<res; j++){

if (res%j == 0){

System.out.print(" "+j);

}

}

System.out.println("");

}

}

}

static boolean isPerfect(int num){

int sum = 0;

int b = 0;

for (int i = 1; i<num; i++){

if (num%i == 0){

b = i;

sum = sum + b;

}

}

if ( sum == num){

return true;

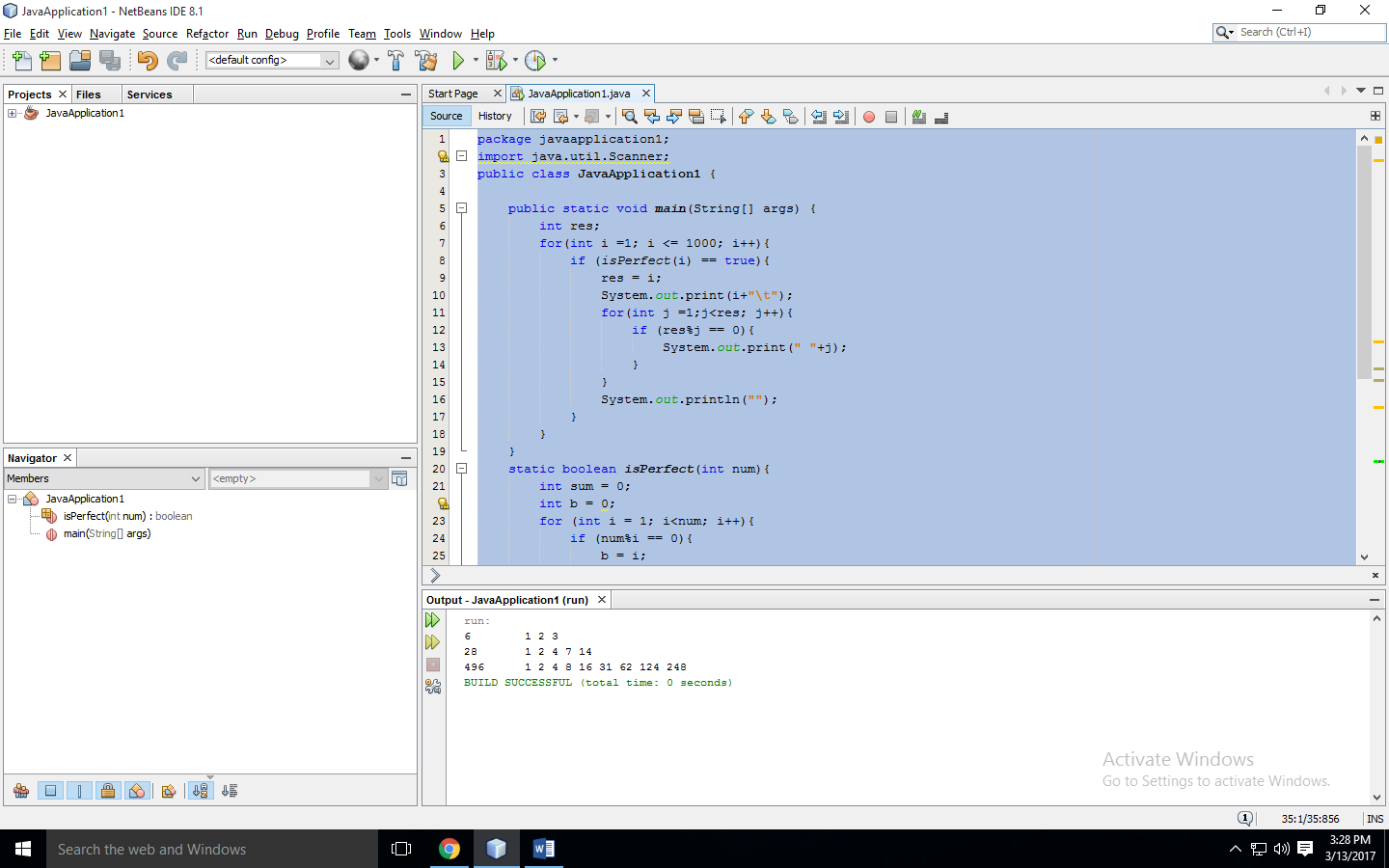
}

return false;

}

}

**Output:**



**Task two:**

package javaapplication1;

import java.util.Scanner;

public class JavaApplication1 {

public static void main(String[] args) {

SavingsAccount saver1 = new SavingsAccount(2000.00);

SavingsAccount saver2 = new SavingsAccount(3000.00);

saver1.modifyIntrestRate(4);

System.out.println("When intrest rate is 4%");

System.out.println("Next months intrest of saver1: "+saver1.calculateMonthlyIntrest());

System.out.println("Next months balance of saver1: "+saver1.getSavingsBalance());

System.out.println("Next months intrest of saver2: "+saver2.calculateMonthlyIntrest());

System.out.println("Next months balance of saver2: "+saver2.getSavingsBalance());

saver1.modifyIntrestRate(5);

System.out.println("When intrest rate is 5%");

System.out.println("Next months intrest of saver1: "+saver1.calculateMonthlyIntrest());

System.out.println("Next months balance of saver1: "+saver1.getSavingsBalance());

System.out.println("Next months intrest of saver2: "+saver2.calculateMonthlyIntrest());

System.out.println("Next months balance of saver2: "+saver2.getSavingsBalance());

}

}

class SavingsAccount{

static double annualIntrestRate;

private double savingsBalance;

SavingsAccount(double balance){

savingsBalance = balance;

}

double calculateMonthlyIntrest(){

double mi = (savingsBalance \* annualIntrestRate)/12;

savingsBalance += mi;

return mi;

}

static void modifyIntrestRate(double rate){

rate /= 100;

annualIntrestRate = rate;

}

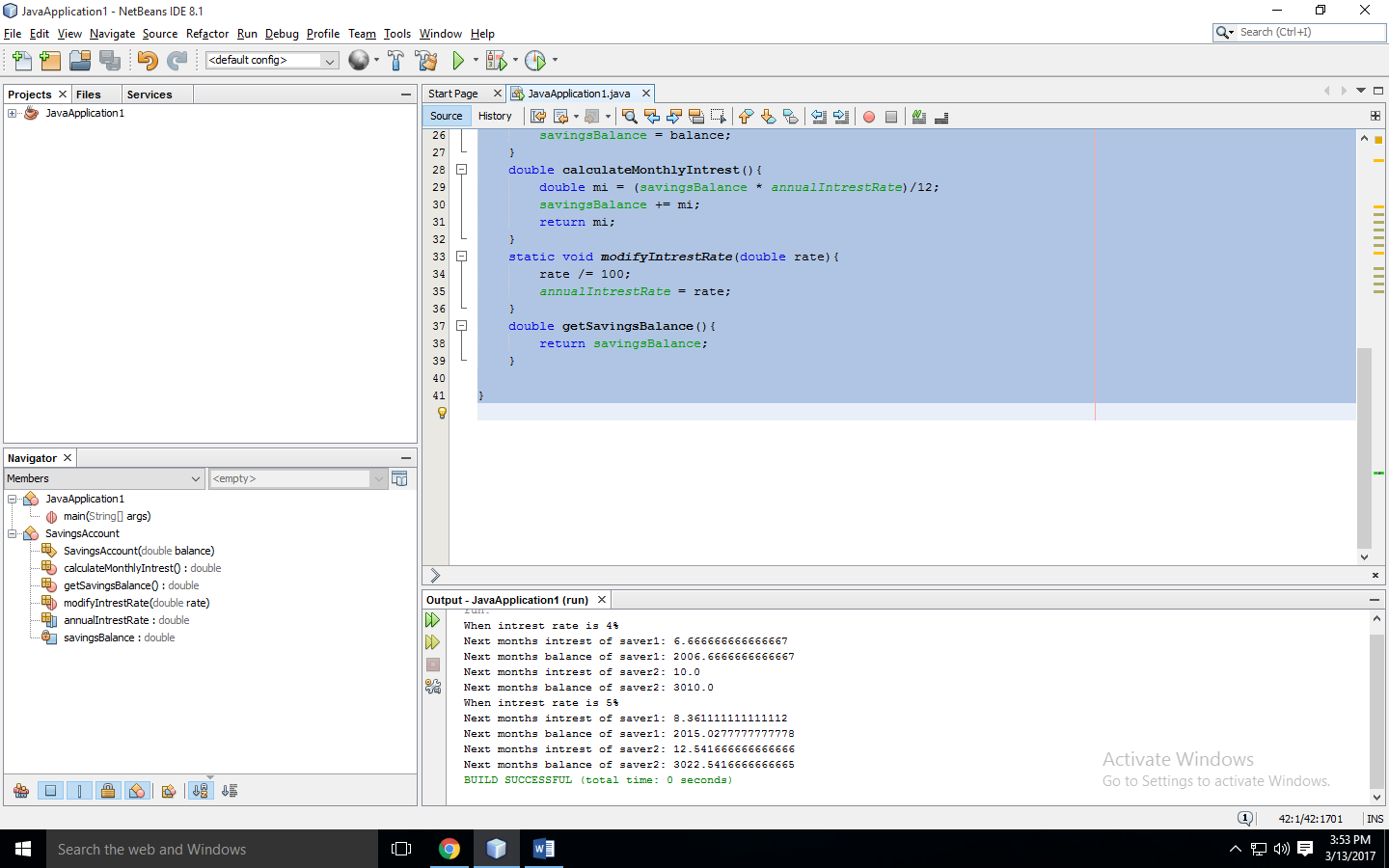
double getSavingsBalance(){

return savingsBalance;

}

}

**Output:**



**Task three:**

package javaapplication1;

import java.util.Scanner;

public class JavaApplication1 {

public static void main(String[] args) {

Time2 t1 = new Time2(11,59,59);

System.out.println("Time before: "+t1.toString());

t1.tickMethod();

System.out.println("Second Increment: "+t1.toString());

t1.incrementMinute();

System.out.println("Minute Increment: "+t1.toString());

t1.incrementHour();

System.out.println("Hour Increment: "+t1.toString());

}

}

class Time2

{

private int hour; // 0 - 23

private int minute; // 0 - 59

private int second; // 0 - 59

// Time2 no-argument constructor:

// initializes each instance variable to zero

public Time2()

{

this( 0, 0, 0 ); // invoke Time2 constructor with three arguments

} // end Time2 no-argument constructor

// Time2 constructor: hour supplied, minute and second defaulted to 0

public Time2( int h )

{

this( h, 0, 0 ); // invoke Time2 constructor with three arguments

} // end Time2 one-argument constructor

// Time2 constructor: hour and minute supplied, second defaulted to 0

public Time2( int h, int m )

{

this( h, m, 0 ); // invoke Time2 constructor with three arguments

} // end Time2 two-argument constructor

// Time2 constructor: hour, minute and second supplied

public Time2( int h, int m, int s )

{

setTime( h, m, s ); // invoke setTime to validate time

} // end Time2 three-argument constructor

// Time2 constructor: another Time2 object supplied

public Time2( Time2 time )

{

// invoke Time2 three-argument constructor

this( time.getHour(), time.getMinute(), time.getSecond() );

} // end Time2 constructor with a Time2 object argument

// Set Methods

// set a new time value using universal time;

// validate the data

public void setTime( int h, int m, int s )

{

setHour( h ); // set the hour

setMinute( m ); // set the minute

setSecond( s ); // set the second

} // end method setTime

// validate and set hour

public void setHour( int h )

{

if ( h >= 0 && h < 24 )

hour = h;

else

throw new IllegalArgumentException( "hour must be 0-23" );

} // end method setHour

// validate and set minute

public void setMinute( int m )

{

if ( m >= 0 && m < 60 )

minute = m;

else

throw new IllegalArgumentException( "minute must be 0-59" );

} // end method setMinute

// validate and set second

public void setSecond( int s )

{

if ( s >= 0 && s < 60 )

second = ( ( s >= 0 && s < 60 ) ? s : 0 );

else

throw new IllegalArgumentException( "second must be 0-59" );

} // end method setSecond

// Get Methods

public int getHour(){

return hour;

}

public int getMinute(){

return minute;

}

public int getSecond(){

return second;

}

public void tickMethod(){

if (second < 59){

setSecond(second+1);

}

else{

setSecond(0);

if(minute < 59){

setMinute(minute+1);

}

else{

setSecond(0);

setMinute(0);

if(hour < 23){

setHour(hour+1);

}

else{

setTime(0,0,0);

}

}

}

}

public void incrementMinute(){

if(minute<59){

setMinute(minute+1);

}

else{

setMinute(0);

if(hour < 23){

setHour(hour+1);

}

else{

setTime(0,0,second);

}

}

}

public void incrementHour(){

if(hour < 23){

setHour(hour+1);

}

else{

setTime(0,minute,second);

}

}

// convert to String in universal-time format (HH:MM:SS)

public String toUniversalString()

{

return String.format("%02d:%02d:%02d", getHour(), getMinute(), getSecond()

);

}

// convert to String in standard-time format (H:MM:SS AM or PM)

public String toString()

{

return String.format( "%d:%02d:%02d %s",

( (getHour() == 0 || getHour() == 12) ? 12 : getHour() % 12 ),

getMinute(), getSecond(), ( getHour() < 12 ? "AM" : "PM" ) );

} // end method toString

} // end class Time2

**Output:**

