ABDUL BASIT

193227

BSCS-6C

LAB\_04

**ACTIVITY #1**

class Time

{

private int hour; // 0 - 23

private int minute; // 0 - 59

private int second; // 0 - 59

// TODO: Add constructors definitions here

Time ()//NO ARGS CONSTRUCTOR

{

setTime( 0, 0, 0 );

}

Time (int h)//ONLY HOUR CONSTRUCTOR

{

setTime( h, 0,0 );

}

Time (int h,int m)//HOUR AND MINUTE CONSTRUCTOR

{

setTime( h, m, 0);

}

Time(int h,int m,int s)//THREE ARGUMENTS CONSTRUCTOR

{

setTime( h, m, s );

}

// 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// end class Time

// This class tests the Time Class by creating different objects.

public class TimeTest{

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, 56); // 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);

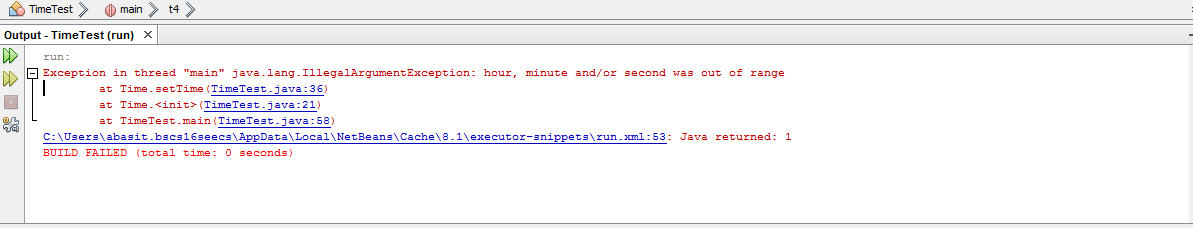
System.out.println(t2);

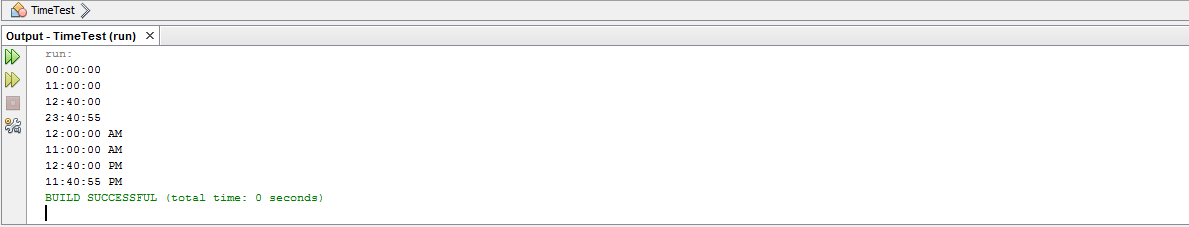
System.out.println(t3);

}

}

**OUTPUT:**

**With wrong arguments pass**

**With correct arguments pass**

**ACTIVITY #2**

// Demonstrate method overloading.

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;

}

}

public class OverloadTest {

public static void main(String args[]) {

Overload ol = new Overload();

double result;

// call all versions of test()

ol.test();

ol.test(10);

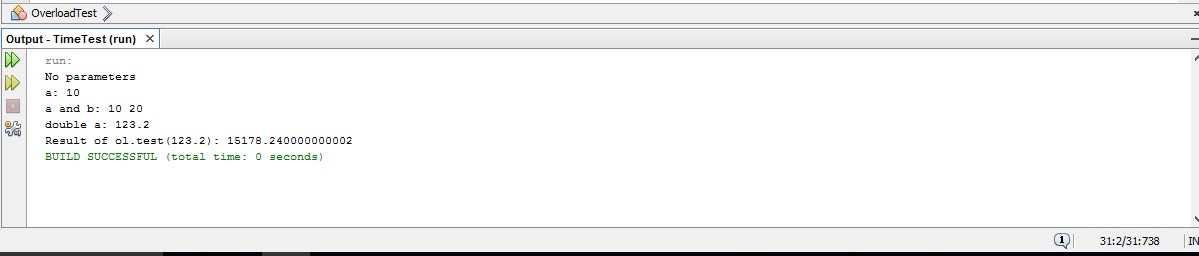
ol.test(10, 20);

result = ol.test(123.2);

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

}

}

**OUTPUT:**

**ACTIVITY #3**

void test(int a) {

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

}

// Overload test for one integer parameter.

int test(int a) {

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

return 0;

}

**The overloading of parametric function is not allowed because repetition of same parameter function is not allowed, although they have different return type but this cannot affect.**

void test(int a, int b) {

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

}

// Overload test for two integer parameters.

void test(int b, int a) {

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

**Here the number of parameters are repeat again so it is not allowed we have to change the type of parameters in order to correct it.**

**TASK #1**

class Task1{//CLASS

//FUNCTION ISPERFECT FOR CHECKING NUMBER VIA BOOLEAN RETURN TYPE

public static boolean isPerfect(int x){

int number=x;

int sum=0;

for(int j=1;j<number;j++){//LOOP FOR CHECKING PERFECT NUMBER

if(number%j==0){

sum+=j;

}

}

if(number==sum)

return true;//RETURNING TRUE IF NUMBER IS PERFECT

else

return false;//RETURNING FALSE IF NUMBER IS NOT PERFECT

}//END ISPERFECT FUNTION

}//END CLASS

//CLASS FOR TESTING TASK1 CLASS

public class Task1Test{

public static void main(String [] args) {

Task1 o1=new Task1();

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

if(o1.isPerfect(i)){

System.out.printf("\n%d is perfect number with factors",i);

for(int j=1;j<i;j++){//LOOP FOR CHECKING FACTORS

if(i%j==0)

System.out.printf(" %d ", j);

}//INNER FOR ENDS

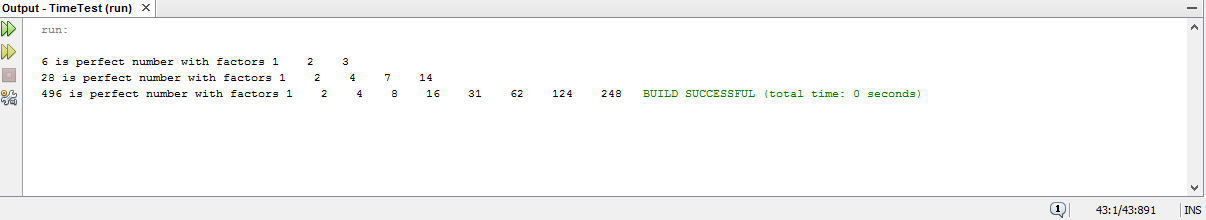
}//IF ENDS

}//OUTER FOR ENDS

}//MAIN ENDS

}//CLASS ENDS

**OUTPUT:**



**TASK#2**

\* @author abasit.bscs16seecs

\*///CLASS

class SavingsAccount {

static double annualInterestRate=0;

private double savingBalance=0;

//CONSTRUCTOR

public SavingsAccount(double iB){

savingBalance=iB;

}

//FUNCTIONS

static void modifyInterestRate(double iR){

annualInterestRate=iR;

}

//INTEREST CALCULATION FUNCTION

public void calculateMonthlyInterest(){

savingBalance=savingBalance + (savingBalance \* annualInterestRate / 12);

}

public double getBalance(){

return savingBalance;

}

}

//TEST CLASS

public class SavingsAccountTest{

public static void main(String args[]) {//MAIN BEGINS

SavingsAccount s1 = new SavingsAccount(2000);

SavingsAccount s2 = new SavingsAccount(3000);

//INTEREST RATE WITH 4% INTEREST

SavingsAccount.modifyInterestRate(.04);

System.out.printf("The initial amounts of Person 1 and Person 2 are \n");

System.out.printf("Person 1 = %.2f\n",s1.getBalance());

System.out.printf("Person 2 = %.2f\n",s2.getBalance());

s1.calculateMonthlyInterest();

s2.calculateMonthlyInterest();

System.out.printf("\nAfter 4%% interest");

System.out.printf("\nAmount of Person 1 is: %.2f ", s1.getBalance());

System.out.printf("\nAmount of Person 2 is: %.2f", s2.getBalance());

//INTEREST RATE WITH 5% INTEREST

SavingsAccount.modifyInterestRate(.05);

s1.calculateMonthlyInterest();

s2.calculateMonthlyInterest();

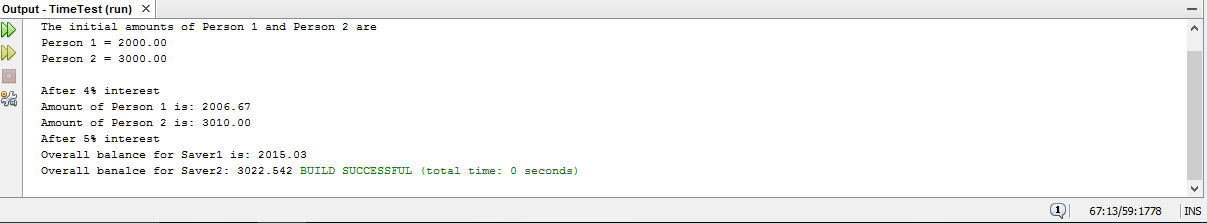
System.out.printf("\nAfter 5%% interest");

System.out.printf("\nOverall balance for Saver1 is: %.2f", s1.getBalance());

System.out.printf("\nOverall banalce for Saver2: %.3f ", s2.getBalance());

}//END MAIN

}//END CLASS

**OUTPUT:**

**TASK#3**

class Time

{

private int hour; // 0 - 2

private int minute; // 0 - 59

private int second; // 0 - 59

//CONCRUCTORS

public Time()

{

this( 0, 0, 0 );

}

public Time( int h )

{

this( h, 0, 0 );

}

public Time( int h, int m )

{

this( h, m, 0 );

}

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

{

setTime( h, m, s );

}

public Time( Time time )

{

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

}

//FUNCTIONS

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

{

setHour( h );

setMinute( m );

setSecond( s );

}

public void setHour( int h )

{

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

hour = h;

else

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

}

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;

}

// 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

public void incrementMinute(){

minute++;

if(minute == 60){

incrementHour();

minute = 0;

}

}

public void incrementHour(){

hour++;

if(hour == 24){

hour = 0;

}

}

public void tick(){

second++;

if(second == 60){

incrementMinute();

second = 0;

}

}

} // END CLASS TIME

public class TimeTest {

public static void main(String[] args){

Time clock = new Time(0,0,0);

for(int second = 0; second <= 86400; second++){

clock.tick();

System.out.println(clock);

}

}

}

