

Week 3 - Lab #1

Section: 103
Simge Tekin

Outline

- Announcements
 - Wednesday: Quiz-1
 - P1 is due on Tuesday
 - P2 will be out on Tuesday
- Questions about P-1 or Eclipse debugger
 - or anything that has been covered in class?
 - Quiz 2 will be about Eclipse debugger
- JUnit (you will need to write tests for Project 2 with assert statements)
- Memory Maps- Recap

JUnit

JUnit is a testing framework for Java

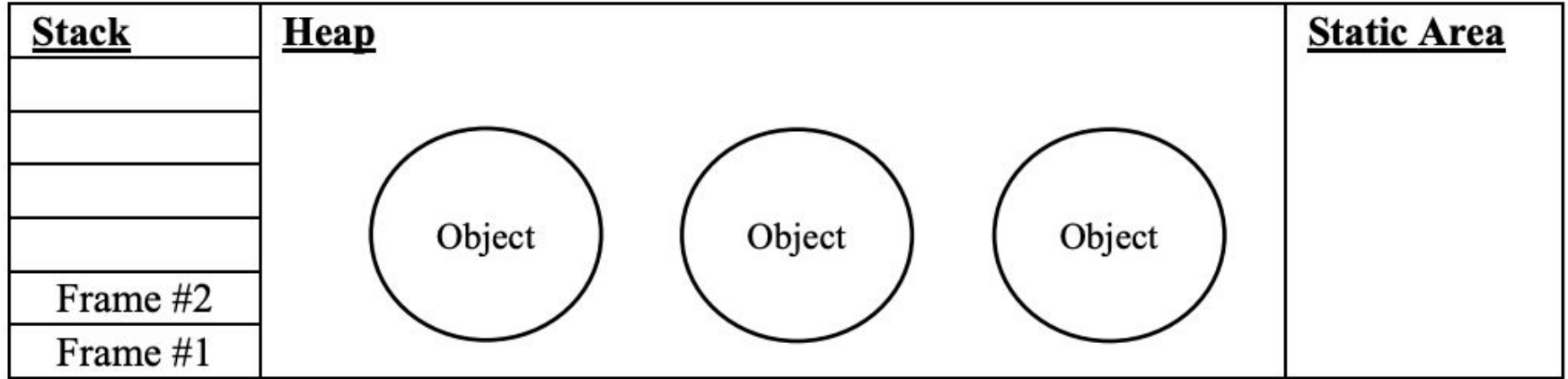
1. **JUnit detects methods marked with `@Test`.**
2. **Runs them automatically.**
3. **Checks assertions (`assertEquals`, `assertTrue`, etc.).**
4. **Reports failures if an assertion fails.**

Let's create some JUnit tests!

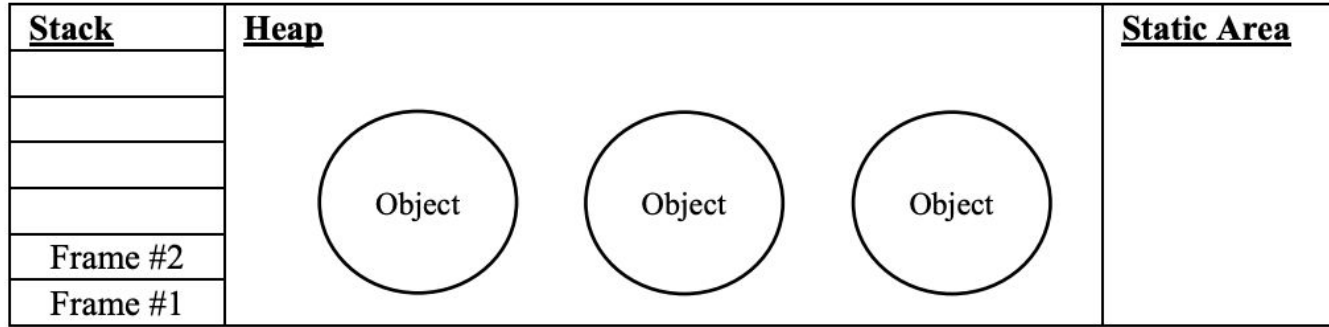
<https://www.cs.umd.edu/~nelson/eclipse/junit/>

- By default, JUnit runs tests in **an arbitrary order**
- You can enforce **name-based ordering** (or other orders) using **MethodSorters**
- **Alphabetical order**

Memory Maps



Memory Maps



Stack: parameters, local variables and the current object reference(if the method is not static)

Heap: Objects (Arrays are strings are objects too!)

Static area: values that are present throughout the duration of the program execution (static variables)

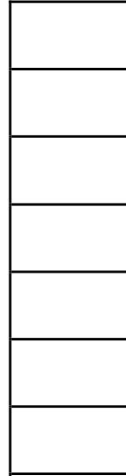
Code area: Where the code resides in memory. We don't draw the code area.

Example 1

Slides from Kamala

```
public class Driver {  
    public static int sumOfSquares(int x, int y) {  
        int answer;  
  
        answer = x * x + y * y;  
  
        /* HERE */  
        return answer;  
    }  
  
    public static void main(String[] args) {  
        int a = 3, b = 4, result;  
  
        result = sumOfSquares(a, b);  
        System.out.println("Answer: " + result);  
    }  
}
```

STACK

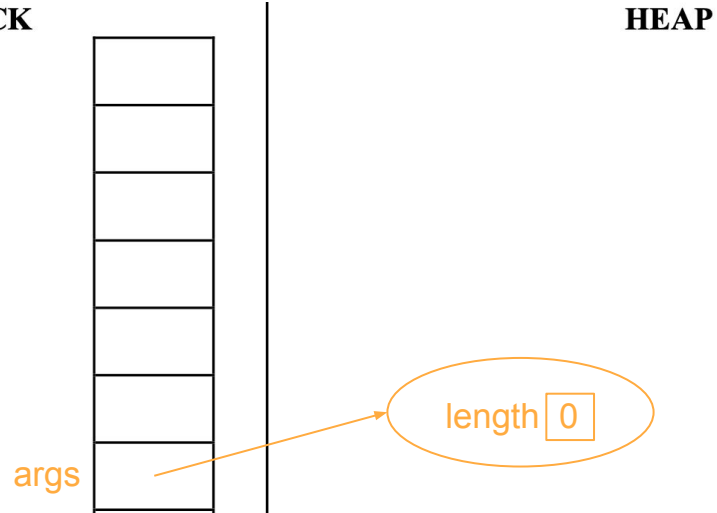


HEAP


STATIC


```
public class Driver {  
    public static int sumOfSquares(int x, int y) {  
        int answer;  
  
        answer = x * x + y * y;  
  
        /* HERE */  
        return answer;  
    }  
  
    public static void main(String[] args) {  
        int a = 3, b = 4, result;  
  
        result = sumOfSquares(a, b);  
        System.out.println("Answer: " + result);  
    }  
}
```

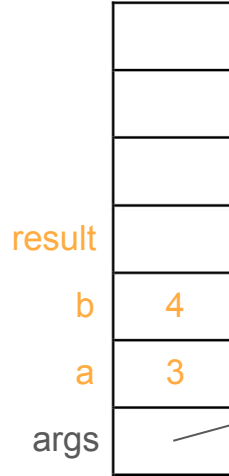
STACK



STATIC

```
public class Driver {  
    public static int sumOfSquares(int x, int y) {  
        int answer;  
  
        answer = x * x + y * y;  
  
        /* HERE */  
        return answer;  
    }  
  
    public static void main(String[] args) {  
        int a = 3, b = 4, result;   
  
        result = sumOfSquares(a, b);  
        System.out.println("Answer: " + result);  
    }  
}
```

STACK



HEAP

STATIC



length 0

```

public class Driver {
    public static int sumOfSquares(int x, int y) {
        int answer;

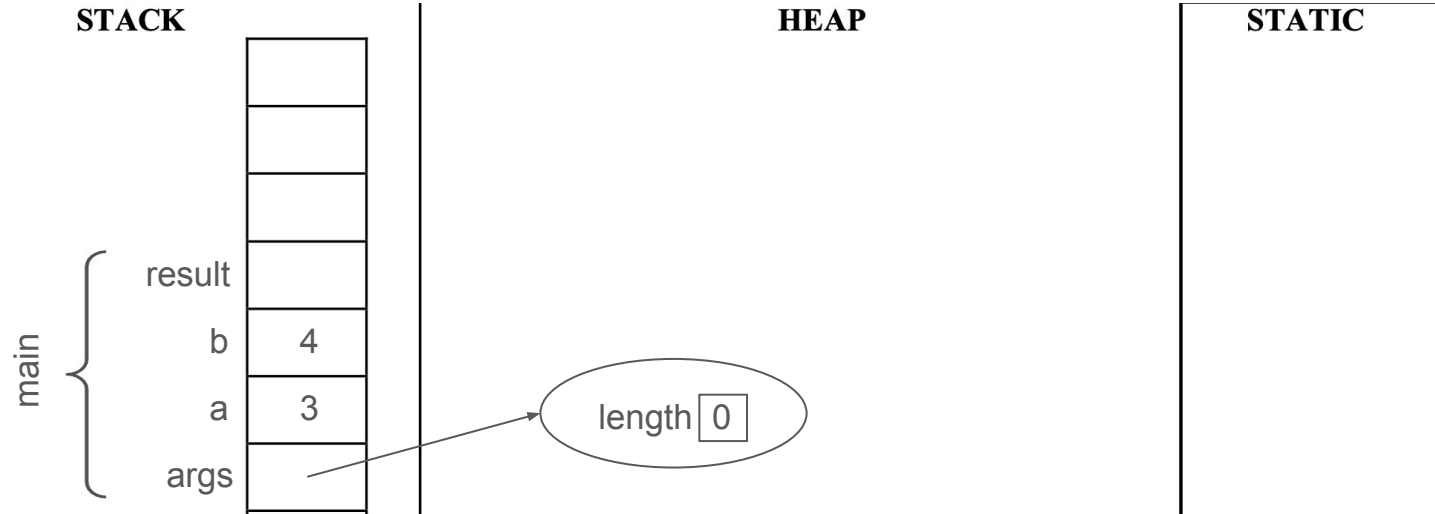
        answer = x * x + y * y;

        /* HERE */
        return answer;
    }

    public static void main(String[] args) {
        int a = 3, b = 4, result;

        result = sumOfSquares(a, b);
        System.out.println("Answer: " + result);
    }
}

```



```

public class Driver {
    public static int sumOfSquares(int x, int y) {
        int answer;

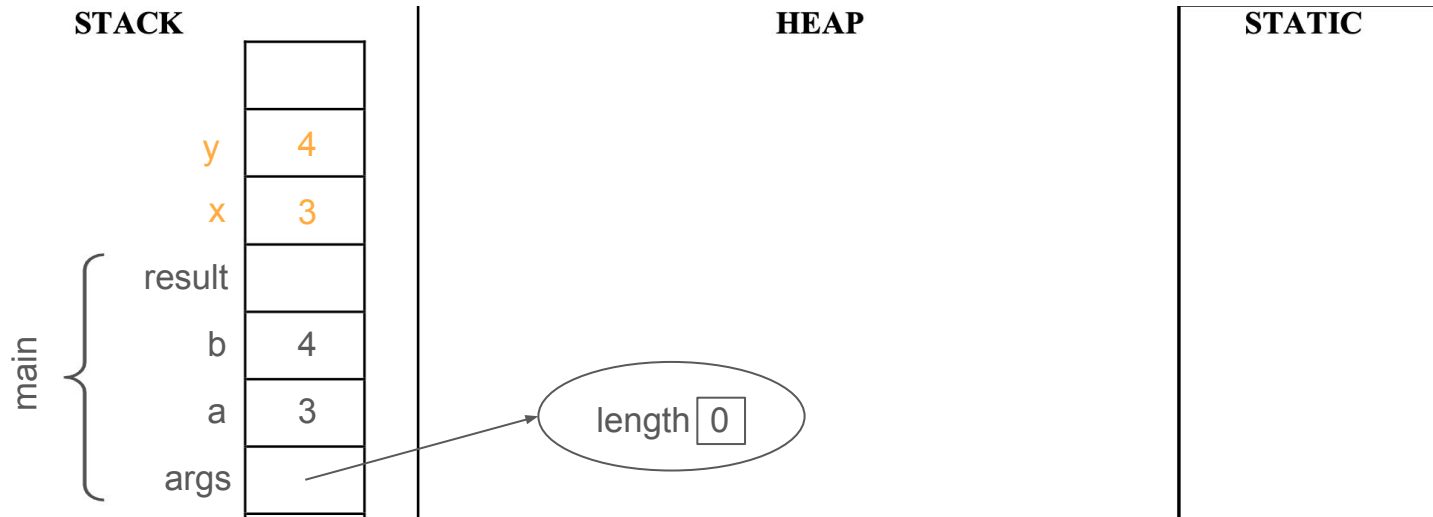
        answer = x * x + y * y;

        /* HERE */
        return answer;
    }

    public static void main(String[] args) {
        int a = 3, b = 4, result;

        result = sumOfSquares(a, b);
        System.out.println("Answer: " + result);
    }
}

```



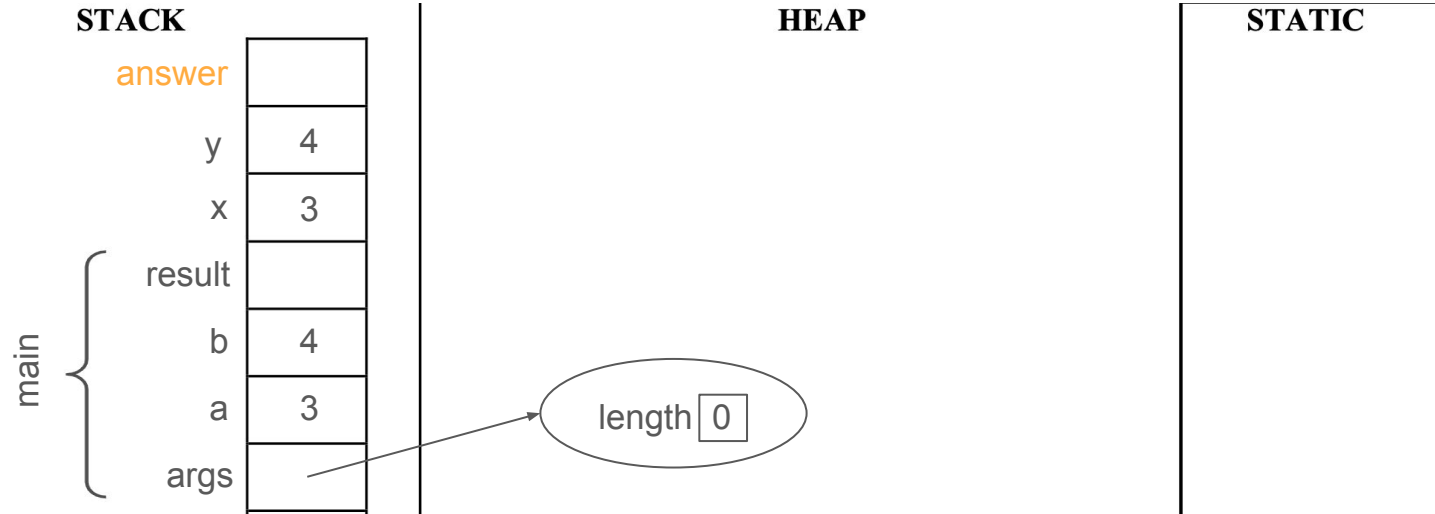
```

public class Driver {
    public static int sumOfSquares(int x, int y) {
        int answer; ←
        answer = x * x + y * y;
        /* HERE */
        return answer;
    }

    public static void main(String[] args) {
        int a = 3, b = 4, result;

        result = sumOfSquares(a, b);
        System.out.println("Answer: " + result);
    }
}

```



```

public class Driver {
    public static int sumOfSquares(int x, int y) {
        int answer;

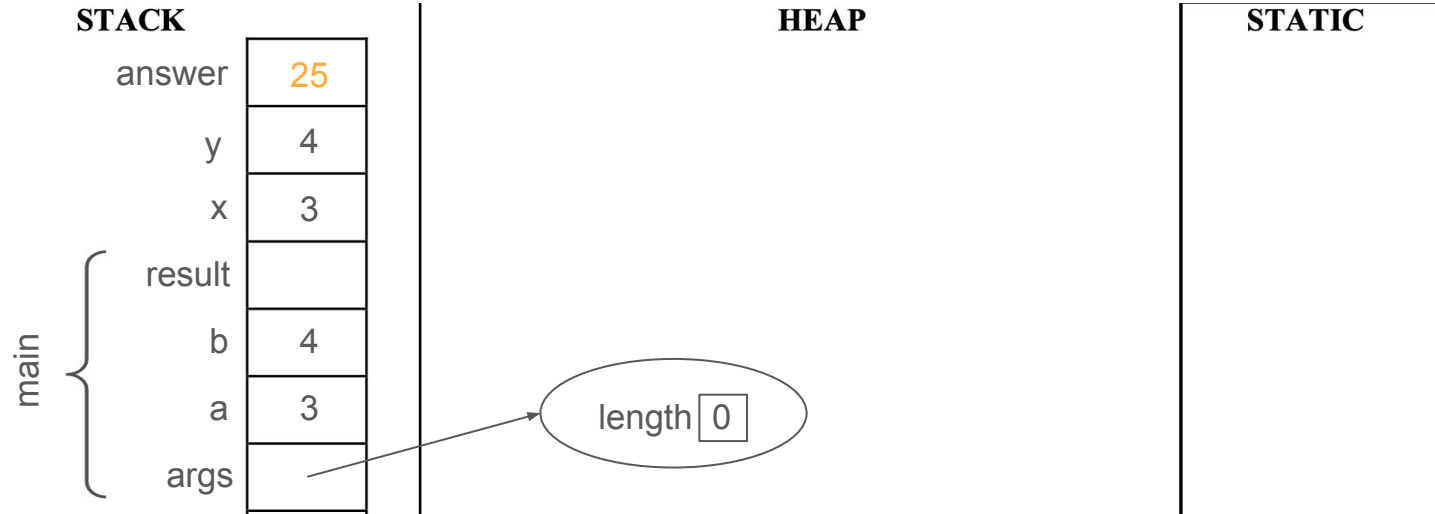
        answer = x * x + y * y;

        /* HERE */
        return answer;
    }

    public static void main(String[] args) {
        int a = 3, b = 4, result;

        result = sumOfSquares(a, b);
        System.out.println("Answer: " + result);
    }
}

```



```

public class Driver {
    public static int sumOfSquares(int x, int y) {
        int answer;

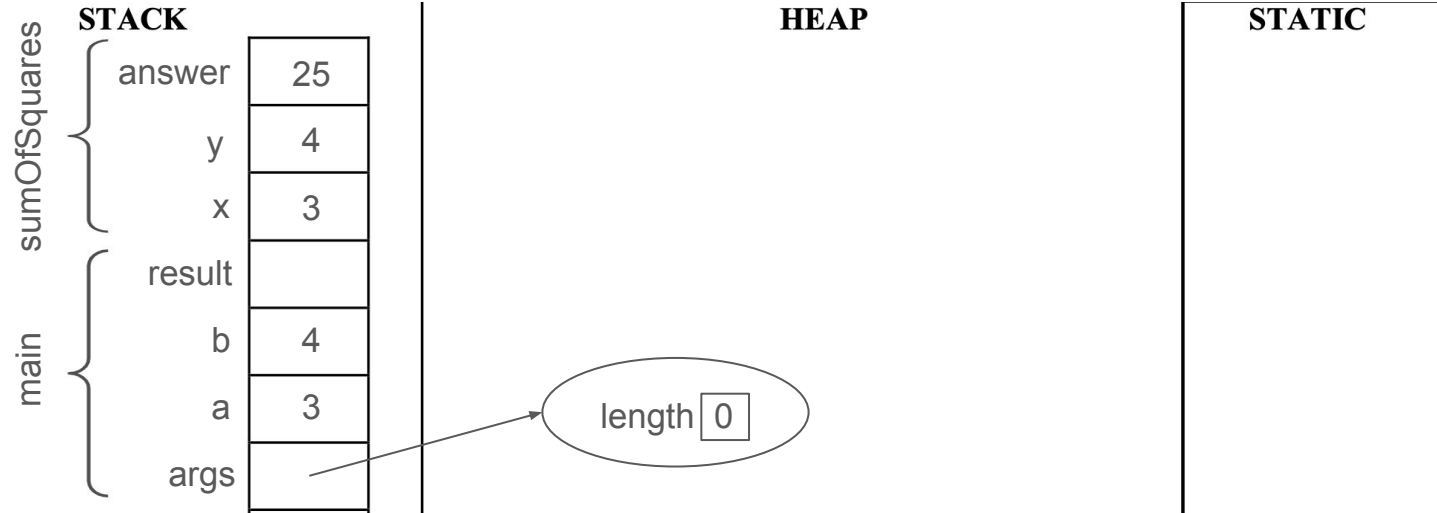
        answer = x * x + y * y;

        /* HERE */
        return answer;
    }

    public static void main(String[] args) {
        int a = 3, b = 4, result;

        result = sumOfSquares(a, b);
        System.out.println("Answer: " + result);
    }
}

```



Example 3


```

public class Person {
    public static double MINIMUM_SALARY = 1000.00;
    private String name;
    private double salary;
    private StringBuffer calls;

    public Person(String name, double salary) {
        this.name = name;
        this.salary = salary;
        calls = new StringBuffer("Calls: ");
    }

    public Person(String name) {
        this(name, MINIMUM_SALARY);
    }

    public Person increaseSalary(double delta) {
        salary += delta;

        /* HERE */
        /* Returning reference to current object */
        return this;
    }

    public void addCall(String newCall) {
        calls.append(newCall);
    }

    public String toString() {
        return name + ", $" + salary + ", " + calls;
    }
}

```

```

public class Driver {

    public static void task(Person friend, double salary) {
        double newSal = salary + 100;
        Person p = friend;

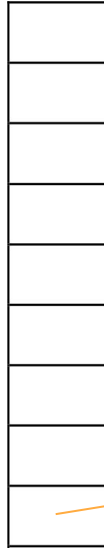
        p.increaseSalary(newSal);
        friend.addCall("toschool");
        friend = null;
    }

    public static void main(String[] args) {
        double pay = 4000;

        Person laura = new Person("Laura", 2000);
        task(laura, pay);
        System.out.println(laura);
    }
}

```

STACK



args

HEAP

STATIC

length 0

```

public class Person {
    public static double MINIMUM_SALARY = 1000.00;
    private String name;
    private double salary;
    private StringBuffer calls;

    public Person(String name, double salary) {
        this.name = name;
        this.salary = salary;
        calls = new StringBuffer("Calls: ");
    }

    public Person(String name) {
        this(name, MINIMUM_SALARY);
    }

    public Person increaseSalary(double delta) {
        salary += delta;

        /* HERE */
        /* Returning reference to current object */
        return this;
    }

    public void addCall(String newCall) {
        calls.append(newCall);
    }

    public String toString() {
        return name + ", $" + salary + ", " + calls;
    }
}

```

```

public class Driver {

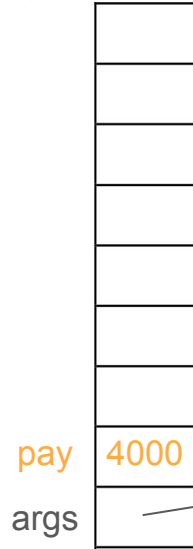
    public static void task(Person friend, double salary) {
        double newSal = salary + 100;
        Person p = friend;

        p.increaseSalary(newSal);
        friend.addCall("toschool");
        friend = null;
    }

    public static void main(String[] args) {
        double pay = 4000;
        Person laura = new Person("Laura", 2000);
        task(laura, pay);
        System.out.println(laura);
    }
}

```

STACK



HEAP

STATIC

```

public class Person {
    public static double MINIMUM_SALARY = 1000.00;
    private String name;
    private double salary;
    private StringBuffer calls;

    public Person(String name, double salary) {
        this.name = name;
        this.salary = salary;
        calls = new StringBuffer("Calls: ");
    }

    public Person(String name) {
        this(name, MINIMUM_SALARY);
    }

    public Person increaseSalary(double delta) {
        salary += delta;

        /* HERE */
        /* Returning reference to current object */
        return this;
    }

    public void addCall(String newCall) {
        calls.append(newCall);
    }

    public String toString() {
        return name + ", $" + salary + ", " + calls;
    }
}

```

```

public class Driver {

    public static void task(Person friend, double salary) {
        double newSal = salary + 100;
        Person p = friend;

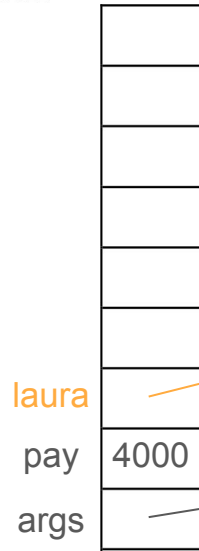
        p.increaseSalary(newSal);
        friend.addCall("toschool");
        friend = null;
    }

    public static void main(String[] args) {
        double pay = 4000;

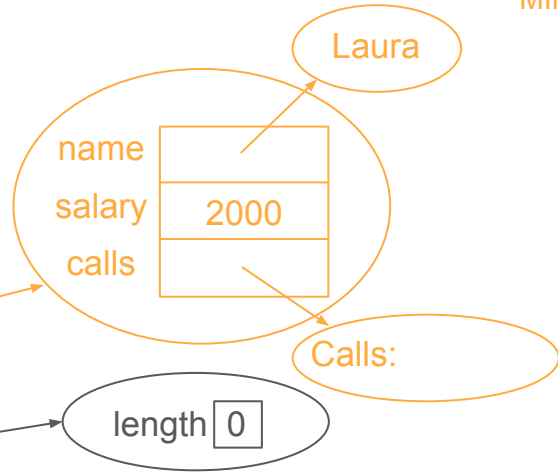
        Person laura = new Person("Laura", 2000);
        task(laura, pay);
        System.out.println(laura);
    }
}

```

STACK



HEAP



STATIC

MINIMUM_SALARY
1000.00

```

public class Person {
    public static double MINIMUM_SALARY = 1000.00;
    private String name;
    private double salary;
    private StringBuffer calls;

    public Person(String name, double salary) {
        this.name = name;
        this.salary = salary;
        calls = new StringBuffer("Calls: ");
    }

    public Person(String name) {
        this(name, MINIMUM_SALARY);
    }

    public Person increaseSalary(double delta) {
        salary += delta;

        /* HERE */
        /* Returning reference to current object */
        return this;
    }

    public void addCall(String newCall) {
        calls.append(newCall);
    }

    public String toString() {
        return name + ", $" + salary + ", " + calls;
    }
}

```

```

public class Driver {

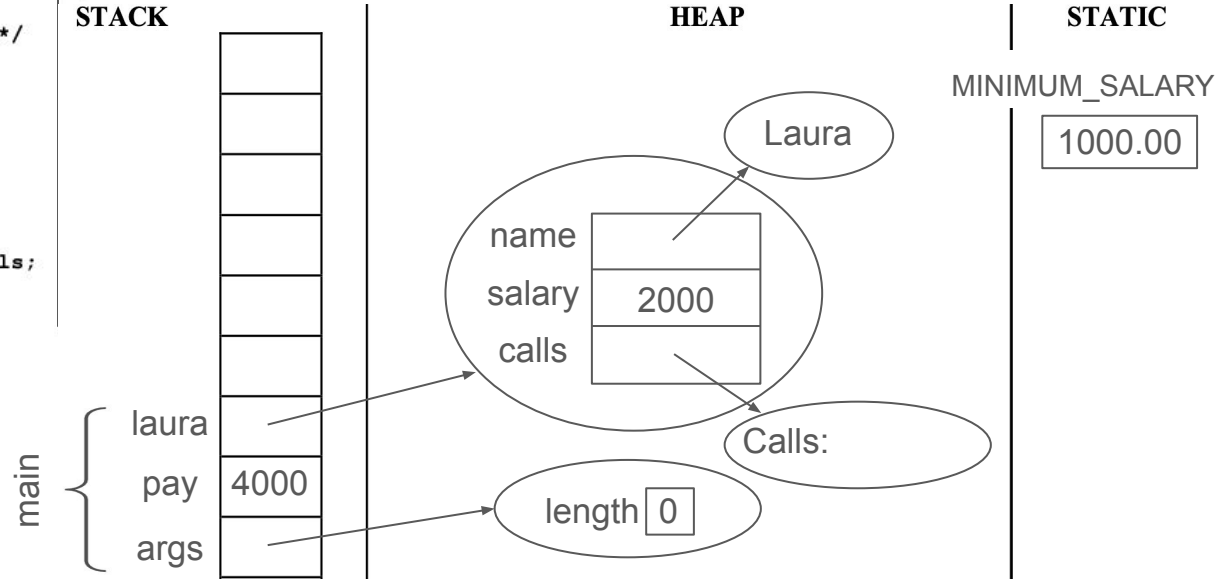
    public static void task(Person friend, double salary) {
        double newSal = salary + 100;
        Person p = friend;

        p.increaseSalary(newSal);
        friend.addCall("toschool");
        friend = null;
    }

    public static void main(String[] args) {
        double pay = 4000;

        Person laura = new Person("Laura", 2000);
        task(laura, pay);
        System.out.println(laura);
    }
}

```



```

public class Person {
    public static double MINIMUM_SALARY = 1000.00;
    private String name;
    private double salary;
    private StringBuffer calls;

    public Person(String name, double salary) {
        this.name = name;
        this.salary = salary;
        calls = new StringBuffer("Calls: ");
    }

    public Person(String name) {
        this(name, MINIMUM_SALARY);
    }

    public Person increaseSalary(double delta) {
        salary += delta;

        /* HERE */
        /* Returning reference to current object */
        return this;
    }

    public void addCall(String newCall) {
        calls.append(newCall);
    }

    public String toString() {
        return name + ", $" + salary + ", " + calls;
    }
}

```

```

public class Driver {

    public static void task(Person friend, double salary) {
        double newSal = salary + 100;
        Person p = friend;

        p.increaseSalary(newSal);
        friend.addCall("toschool");
        friend = null;
    }

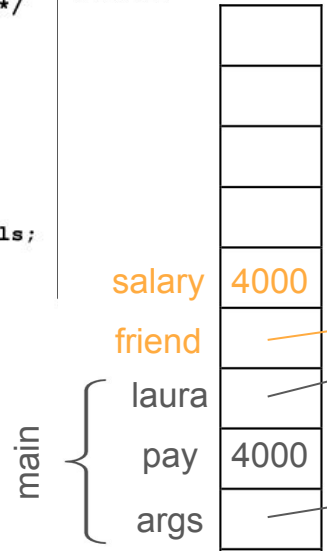
    public static void main(String[] args) {
        double pay = 4000;

        Person laura = new Person("Laura", 2000);
        task(laura, pay);
        System.out.println(laura);
    }
}

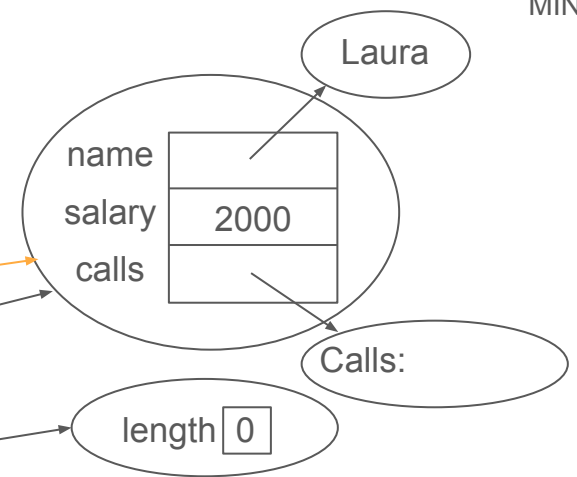
```



STACK



HEAP



STATIC



```

public class Person {
    public static double MINIMUM_SALARY = 1000.00;
    private String name;
    private double salary;
    private StringBuffer calls;

    public Person(String name, double salary) {
        this.name = name;
        this.salary = salary;
        calls = new StringBuffer("Calls: ");
    }

    public Person(String name) {
        this(name, MINIMUM_SALARY);
    }

    public Person increaseSalary(double delta) {
        salary += delta;

        /* HERE */
        /* Returning reference to current object */
        return this;
    }

    public void addCall(String newCall) {
        calls.append(newCall);
    }

    public String toString() {
        return name + ", $" + salary + ", " + calls;
    }
}

```

```

public class Driver {

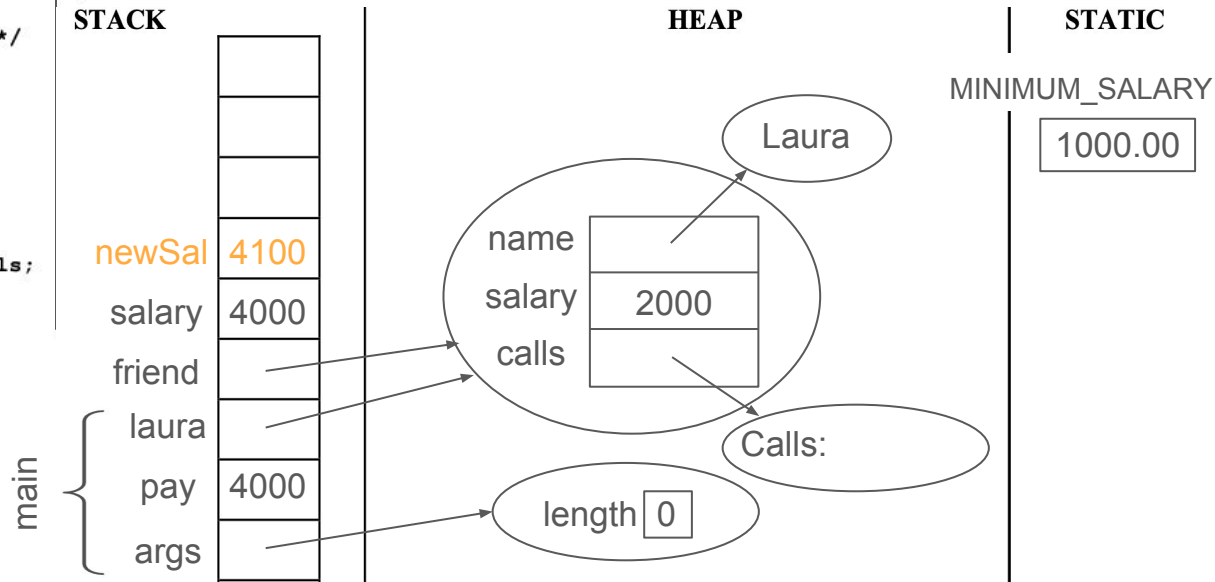
    public static void task(Person friend, double salary) {
        double newSal = salary + 100;
        Person p = friend;

        p.increaseSalary(newSal);
        friend.addCall("toschool");
        friend = null;
    }

    public static void main(String[] args) {
        double pay = 4000;

        Person laura = new Person("Laura", 2000);
        task(laura, pay);
        System.out.println(laura);
    }
}

```



```

public class Person {
    public static double MINIMUM_SALARY = 1000.00;
    private String name;
    private double salary;
    private StringBuffer calls;

    public Person(String name, double salary) {
        this.name = name;
        this.salary = salary;
        calls = new StringBuffer("Calls: ");
    }

    public Person(String name) {
        this(name, MINIMUM_SALARY);
    }

    public Person increaseSalary(double delta) {
        salary += delta;

        /* HERE */
        /* Returning reference to current object */
        return this;
    }

    public void addCall(String newCall) {
        calls.append(newCall);
    }

    public String toString() {
        return name + ", $" + salary + ", " + calls;
    }
}

```

```

public class Driver {

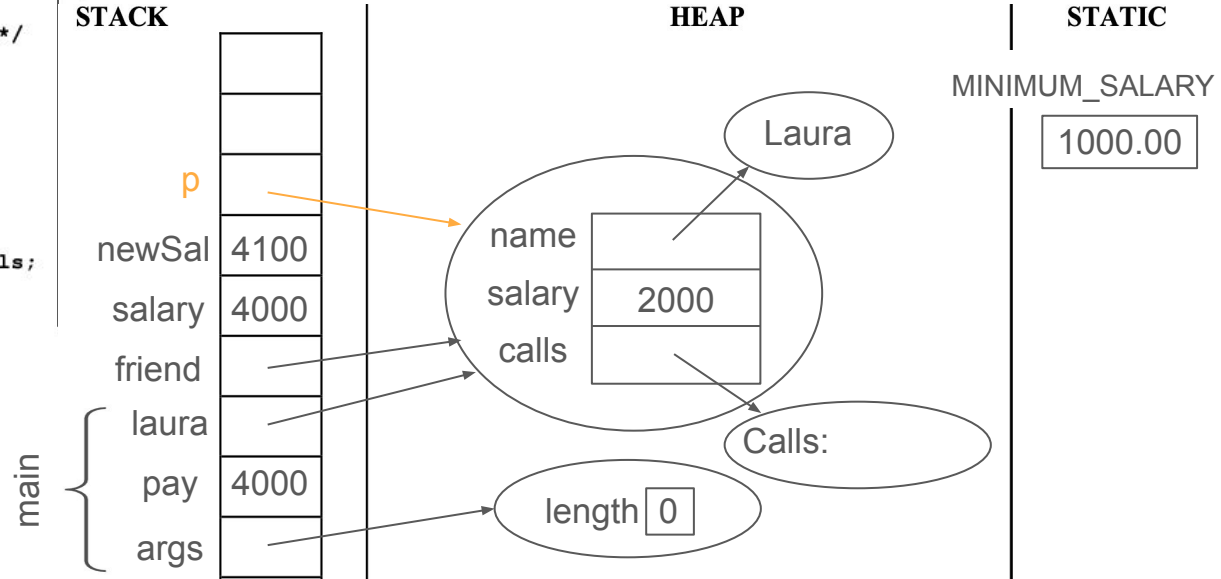
    public static void task(Person friend, double salary) {
        double newSal = salary + 100;
        Person p = friend;

        p.increaseSalary(newSal);
        friend.addCall("toschool");
        friend = null;
    }

    public static void main(String[] args) {
        double pay = 4000;

        Person laura = new Person("Laura", 2000);
        task(laura, pay);
        System.out.println(laura);
    }
}

```




```

public class Person {
    public static double MINIMUM_SALARY = 1000.00;
    private String name;
    private double salary;
    private StringBuffer calls;

    public Person(String name, double salary) {
        this.name = name;
        this.salary = salary;
        calls = new StringBuffer("Calls: ");
    }

    public Person(String name) {
        this(name, MINIMUM_SALARY);
    }

    public Person increaseSalary(double delta) {
        salary += delta;

        /* HERE */
        /* Returning reference to current object */
        return this;
    }

    public void addCall(String newCall) {
        calls.append(newCall);
    }

    public String toString() {
        return name + ", $" + salary + ", " + cal
    }
}

```

```

public class Driver {

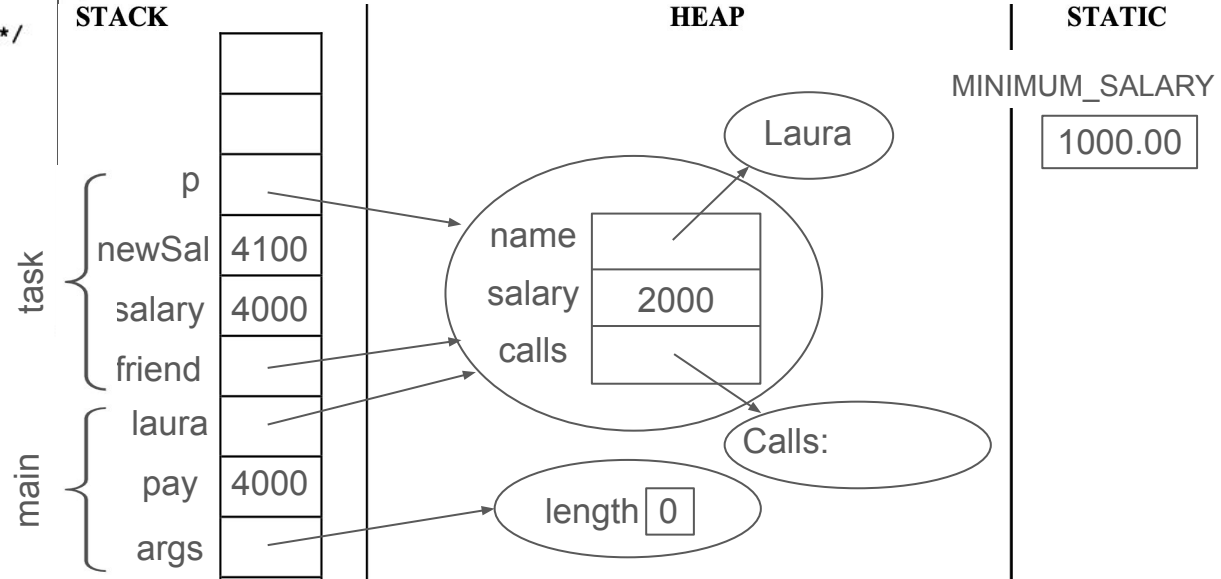
    public static void task(Person friend, double salary) {
        double newSal = salary + 100;
        Person p = friend;

        p.increaseSalary(newSal);
        friend.addCall("toschool");
        friend = null;
    }

    public static void main(String[] args) {
        double pay = 4000;

        Person laura = new Person("Laura", 2000);
        task(laura, pay);
        System.out.println(laura);
    }
}

```




```

public class Person {
    public static double MINIMUM_SALARY = 1000.00;
    private String name;
    private double salary;
    private StringBuffer calls;

    public Person(String name, double salary) {
        this.name = name;
        this.salary = salary;
        calls = new StringBuffer("Calls: ");
    }

    public Person(String name) {
        this(name, MINIMUM_SALARY);
    }

    public Person increaseSalary(double delta) {
        salary += delta;

        /* HERE */
        /* Returning reference to current object */
        return this;
    }

    public void addCall(String newCall) {
        calls.append(newCall);
    }

    public String toString() {
        return name + ", $" + salary + ", " + cal
    }
}

```

```

public class Driver {

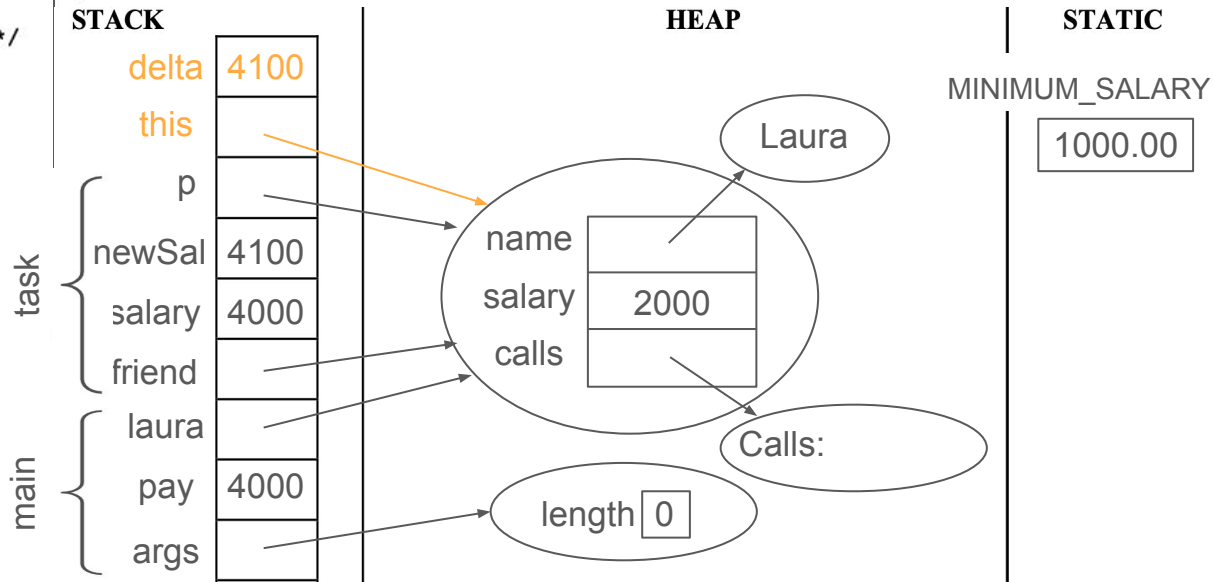
    public static void task(Person friend, double salary) {
        double newSal = salary + 100;
        Person p = friend;

        p.increaseSalary(newSal);
        friend.addCall("toschool");
        friend = null;
    }

    public static void main(String[] args) {
        double pay = 4000;

        Person laura = new Person("Laura", 2000);
        task(laura, pay);
        System.out.println(laura);
    }
}

```



```

public class Person {
    public static double MINIMUM_SALARY = 1000.00;
    private String name;
    private double salary;
    private StringBuffer calls;

    public Person(String name, double salary) {
        this.name = name;
        this.salary = salary;
        calls = new StringBuffer("Calls: ");
    }

    public Person(String name) {
        this(name, MINIMUM_SALARY);
    }

    public Person increaseSalary(double delta) {
        salary += delta;

        /* HERE */
        /* Returning reference to current object */
        return this;
    }

    public void addCall(String newCall) {
        calls.append(newCall);
    }

    public String toString() {
        return name + ", $" + salary + ", " + cal
    }
}

```

```

public class Driver {

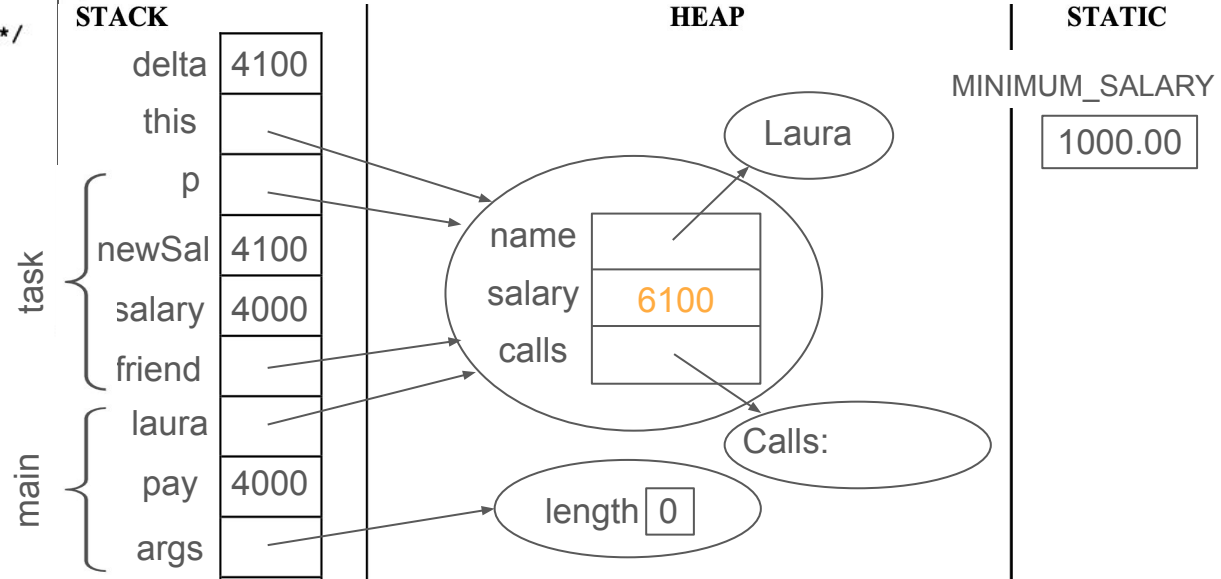
    public static void task(Person friend, double salary) {
        double newSal = salary + 100;
        Person p = friend;

        p.increaseSalary(newSal);
        friend.addCall("toschool");
        friend = null;
    }

    public static void main(String[] args) {
        double pay = 4000;

        Person laura = new Person("Laura", 2000);
        task(laura, pay);
        System.out.println(laura);
    }
}

```



```

public class Person {
    public static double MINIMUM_SALARY = 1000.00;
    private String name;
    private double salary;
    private StringBuffer calls;

    public Person(String name, double salary) {
        this.name = name;
        this.salary = salary;
        calls = new StringBuffer("Calls: ");
    }

    public Person(String name) {
        this(name, MINIMUM_SALARY);
    }

    public Person increaseSalary(double delta) {
        salary += delta;

        /* HERE */
        /* Returning reference to current object
        return this;
    }

    public void addCall(String newCall) {
        calls.append(newCall);
    }

    public String toString() {
        return name + ", $" + salary + ", " + cal
    }
}

```

```

public class Driver {

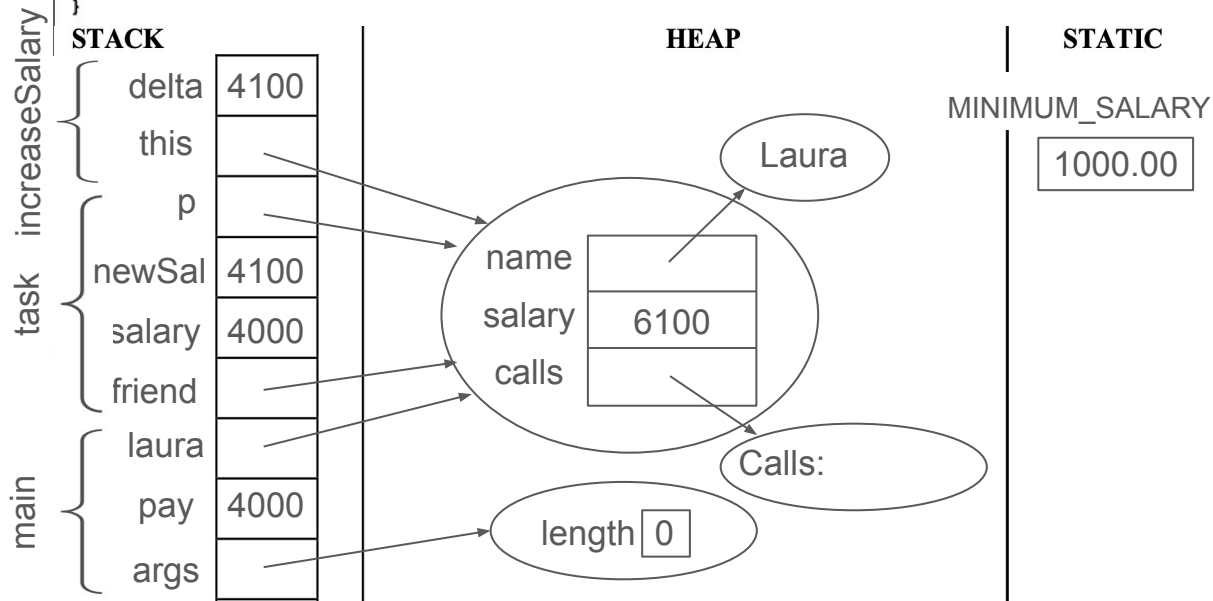
    public static void task(Person friend, double salary) {
        double newSal = salary + 100;
        Person p = friend;

        p.increaseSalary(newSal);
        friend.addCall("toschool");
        friend = null;
    }

    public static void main(String[] args) {
        double pay = 4000;

        Person laura = new Person("Laura", 2000);
        task(laura, pay);
        System.out.println(laura);
    }
}

```



```

public class Person {
    public static double MINIMUM_SALARY = 1000.00;
    private String name;
    private double salary;
    private StringBuffer calls;

    public Person(String name, double salary) {
        this.name = name;
        this.salary = salary;
        calls = new StringBuffer("Calls: ");
    }

    public Person(String name) {
        this(name, MINIMUM_SALARY);
    }

    public Person increaseSalary(double delta) {
        salary += delta;

        /* HERE */
        /* Returning reference to current object
        return this;
    }

    public void addCall(String newCall) {
        calls.append(newCall);
    }

    public String toString() {
        return name + ", $" + salary + ", " + cal
    }
}

```

```

public class Driver {

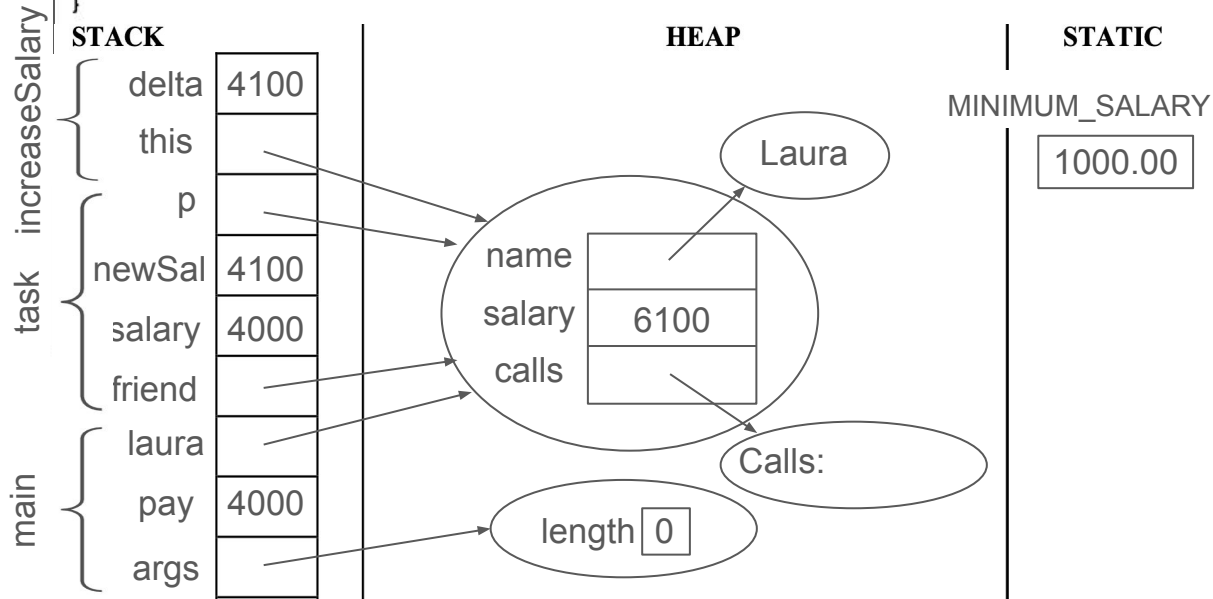
    public static void task(Person friend, double salary) {
        double newSal = salary + 100;
        Person p = friend;

        p.increaseSalary(newSal);
        friend.addCall("toschool");
        friend = null;
    }

    public static void main(String[] args) {
        double pay = 4000;

        Person laura = new Person("Laura", 2000);
        task(laura, pay);
        System.out.println(laura);
    }
}

```



Example 2

```

public class Person {
    public static double MINIMUM_SALARY = 1000.00;
    private String name;
    private double salary;
    private StringBuffer calls;

    public Person(String name, double salary) {
        this.name = name;
        this.salary = salary;
        calls = new StringBuffer("Calls: ");
    }

    public Person(String name) {
        this(name, MINIMUM_SALARY);
    }

    public Person increaseSalary(double delta) {
        salary += delta;

        /* Returning reference to current object */
        return this;
    }

    public void addCall(String newCall) {
        calls.append(newCall);
    }

    public String toString() {
        return name + ", $" + salary + ", " + calls;
    }
}

```

```

public class Driver {

    public static void task(Person friend, double salary) {
        double newSal = salary + 100;
        Person p = friend;

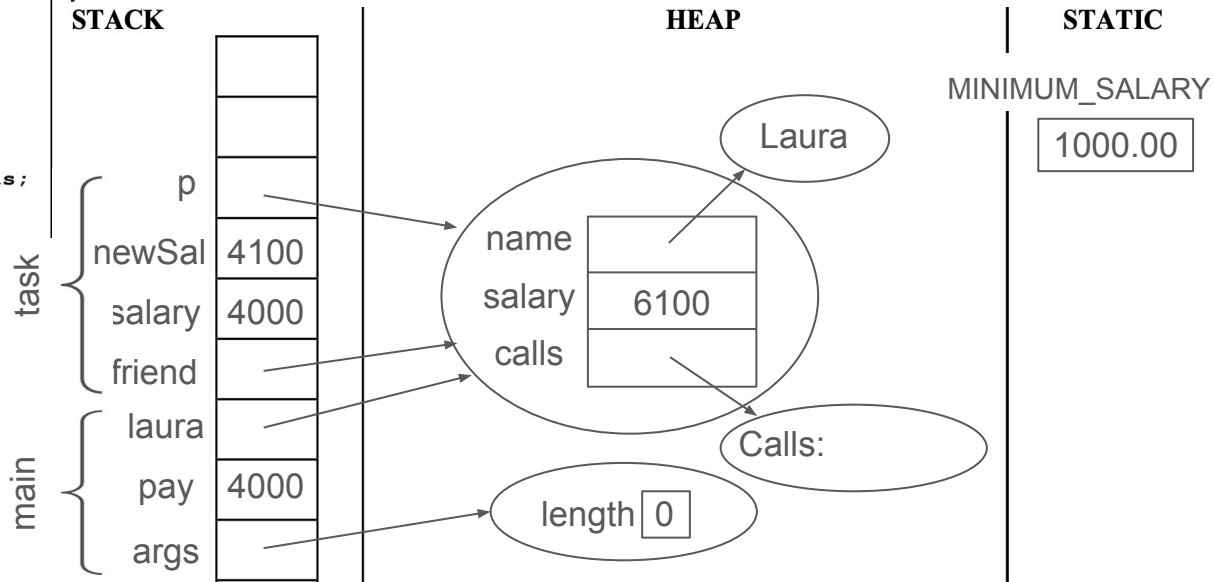
        p.increaseSalary(newSal);
        friend.addCall("toschool");
        friend = null;

        /* HERE */
    }

    public static void main(String[] args) {
        double pay = 4000;

        Person laura = new Person("Laura", 2000);
        task(laura, pay);
        System.out.println(laura);
    }
}

```



```

public class Person {
    public static double MINIMUM_SALARY = 1000.00;
    private String name;
    private double salary;
    private StringBuffer calls;

    public Person(String name, double salary) {
        this.name = name;
        this.salary = salary;
        calls = new StringBuffer("Calls: ");
    }

    public Person(String name) {
        this(name, MINIMUM_SALARY);
    }

    public Person increaseSalary(double delta) {
        salary += delta;

        /* Returning reference to current object */
        return this;
    }

    public void addCall(String newCall) {
        calls.append(newCall);
    }

    public String toString() {
        return name + ", $" + salary + ", " + calls;
    }
}

```

```

public class Driver {

    public static void task(Person friend, double salary) {
        double newSal = salary + 100;
        Person p = friend;

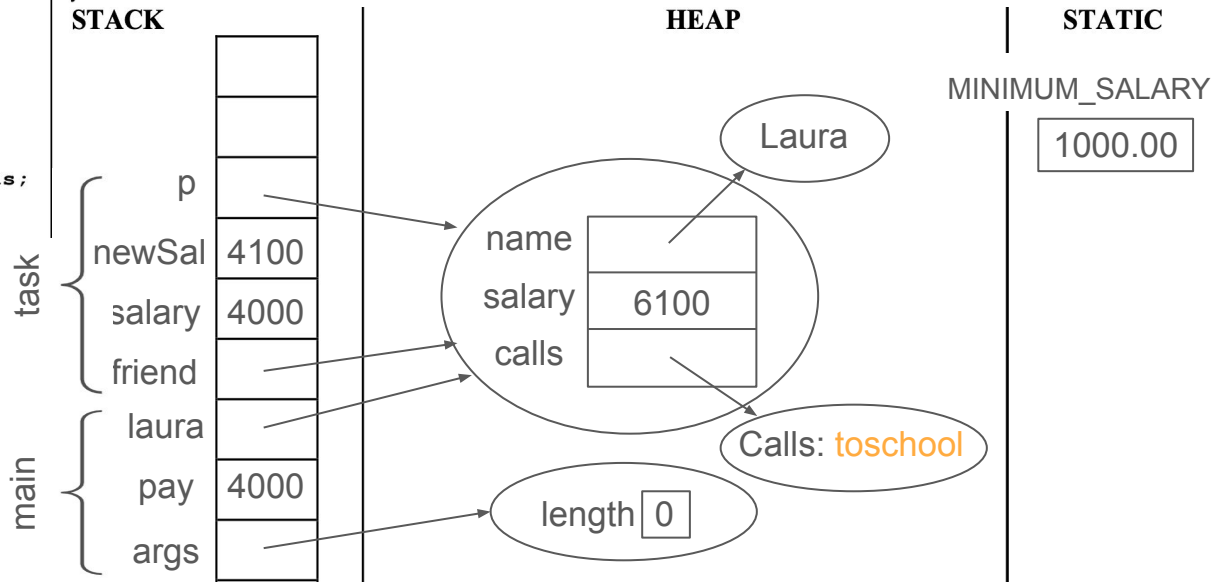
        p.increaseSalary(newSal);
        friend.addCall("toschool");
        friend = null;

        /* HERE */
    }

    public static void main(String[] args) {
        double pay = 4000;

        Person laura = new Person("Laura", 2000);
        task(laura, pay);
        System.out.println(laura);
    }
}

```



```

public class Person {
    public static double MINIMUM_SALARY = 1000.00;
    private String name;
    private double salary;
    private StringBuffer calls;

    public Person(String name, double salary) {
        this.name = name;
        this.salary = salary;
        calls = new StringBuffer("Calls: ");
    }

    public Person(String name) {
        this(name, MINIMUM_SALARY);
    }

    public Person increaseSalary(double delta) {
        salary += delta;

        /* Returning reference to current object */
        return this;
    }

    public void addCall(String newCall) {
        calls.append(newCall);
    }

    public String toString() {
        return name + ", $" + salary + ", " + calls;
    }
}

```

```

public class Driver {

    public static void task(Person friend, double salary) {
        double newSal = salary + 100;
        Person p = friend;

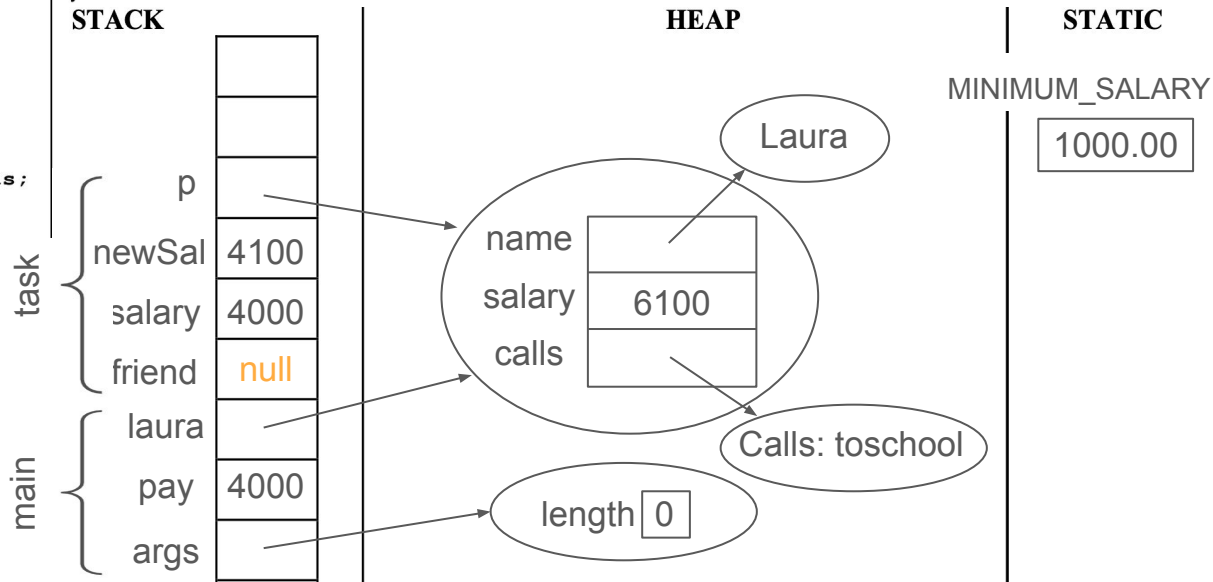
        p.increaseSalary(newSal);
        friend.addCall("toschool");
        friend = null; ←
    }

    /* HERE */

    public static void main(String[] args) {
        double pay = 4000;

        Person laura = new Person("Laura", 2000);
        task(laura, pay);
        System.out.println(laura);
    }
}

```




```

public class Person {
    public static double MINIMUM_SALARY = 1000.00;
    private String name;
    private double salary;
    private StringBuffer calls;

    public Person(String name, double salary) {
        this.name = name;
        this.salary = salary;
        calls = new StringBuffer("Calls: ");
    }

    public Person(String name) {
        this(name, MINIMUM_SALARY);
    }

    public Person increaseSalary(double delta) {
        salary += delta;

        /* Returning reference to current object */
        return this;
    }

    public void addCall(String newCall) {
        calls.append(newCall);
    }

    public String toString() {
        return name + ", $" + salary + ", " + calls;
    }
}

```

```

public class Driver {

    public static void task(Person friend, double salary) {
        double newSal = salary + 100;
        Person p = friend;

        p.increaseSalary(newSal);
        friend.addCall("toschool");
        friend = null;

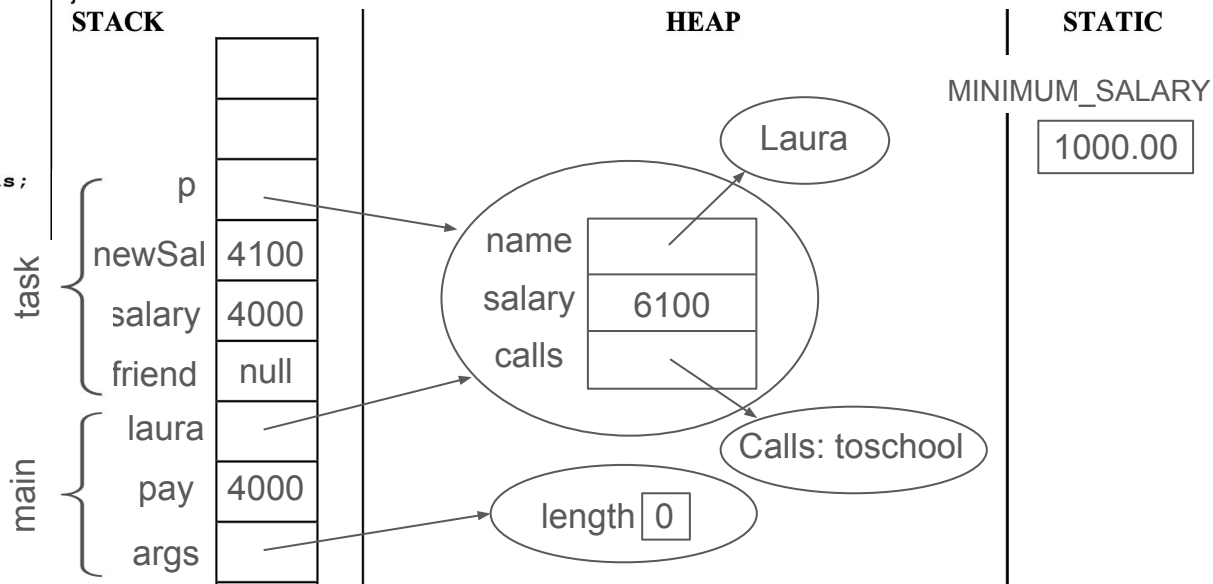
        /* HERE */

    }

    public static void main(String[] args) {
        double pay = 4000;

        Person laura = new Person("Laura", 2000);
        task(laura, pay);
        System.out.println(laura);
    }
}

```



<https://www.cs.umd.edu/class/spring2025/cmsc132-010X-020X/resources/memorymap.html>