LUND
UNIVERSITY

Department of
AUTOMATIC CONTROL

Real-Time Systems

Exam January 3, 2017, hours: 8.00–13.00

Points and grades

All answers must include a clear motivation and a well-formulated answer. Answers may be given in **English or Swedish**. The total number of points is 25. The maximum number of points is specified for each subproblem.

Accepted aid

The textbooks Real-Time Control Systems and Computer Control: An Overview - Educational Version. Standard mathematical tables and authorized "Real-Time Systems Formula Sheet". Pocket calculator.

Results

The result of the exam will become accessible through LADOK. The solutions will be available on WWW:
http://www.control.lth.se/course/FRTN01/

Solutions to the exam in Real-Time Systems 170103

These solutions are available on WWW:
http://www.control.lth.se/course/FRTN01/

1. A system is described by the difference equation

$$y(k+2) - 2.5y(k+1) + y(k) = u(k+1) + 0.5u(k)$$

a. Determine for the system the pulse transfer function from u to y. (1 p)

b. Is the system stable? Motivate your answer. (0.5 p)

Solution

a. Applying the Z-transform on the system gives

$$(z^2 - 2.5z + 1)Y(z) = (z + 0.5)U(z) \Leftrightarrow Y(z) = \frac{z + 0.5}{z^2 - 2.5z + 1}U(z)$$

b. The roots of the transfer function are 0.5 and 2. Since one of the poles of the system is outside the unit circle, the system is unstable.

2. Consider the system

$$\begin{aligned} x(k+1) &= \begin{bmatrix} 0.5 & 0 \\ 1 & 0.9 \end{bmatrix} x(k) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(k) \\ y(k) &= \begin{bmatrix} 0 & 1 \end{bmatrix} x(k) \end{aligned}$$

a. Design a state feedback controller such as the poles of the closed loop systems are both situated in 0.5 and the unit static gain is equal to 1. (2 p)

b. Design an observer on predictor form with both poles at $z = 1/4$. To get the full points, you should also write down the equations for the observer. (2 p)

Solution

a. The control law for state feedback is $u = l_r r(k) - Lx(k)$ where $L = [l_1, l_2]$. The poles of the closed loop system are given by the characteristic polynomial

$$\begin{aligned} \det(zI - (\Phi - \Gamma L)) &= \det\left(\begin{bmatrix} z & 0 \\ 0 & z \end{bmatrix} - \begin{bmatrix} 0.5 & 0 \\ 1 & 0.9 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ l_1 & l_2 \end{bmatrix}\right) = \\ & (z - 0.5)(z + l_2 - 0.9) \end{aligned}$$

from which it is evident that one pole is already in 0.5. The position of the second pole depends on the value of l_2 and to place it in 0.5 simply means selecing

$$l_2 - 0.9 = -0.5 \rightarrow l_2 = 0.4$$

while the value of l_1 is irrelevant.

Finally, in order to have a static gain of 1

$$l_r = \frac{1}{C(I - \Phi + \Gamma L)^{-1}\Gamma} = l_2 + 0.1 = 0.5$$

b. An observer is given by

$$\hat{x}(k+1) = \Phi\hat{x}(k) + \Gamma u(k) + K(y(k) - C\hat{x}(k)) \rightarrow$$
$$\hat{x}(k+1) = (\Phi - KC)\hat{x}(k) + \Gamma u(k) + Ky(k)$$

where $K = [k_1, k_2]^T$. The poles of the observer are determined by the eigenvalues of $\Phi - KC$, which are computed as

$$\begin{aligned}\det(zI - (\Phi - KC)) &= \det\left(\begin{bmatrix} z & 0 \\ 0 & z \end{bmatrix} - \begin{bmatrix} 0.5 & 0 \\ 1 & 0.9 \end{bmatrix} + \begin{bmatrix} 0 & k_1 \\ 0 & k_2 \end{bmatrix}\right) = \\ & z^2 + (k_2 - 1.4)z + (0.45 - 0.5k_2 + k_1)\end{aligned}$$

and imposing that the poles are in 0.25 means imposing that the characteristic polynomial has the form $(z - 0.25)(z - 0.25)$ which is $z - 0.5z + 0.0625$.

$$z^2 + (k_2 - 1.4)z + (0.45 - 0.5k_2 + k_1) = z - 0.5z + 0.0625$$

gives us

$$k_2 - 1.4 = -0.5 \rightarrow k_2 = 0.9$$

and consequently

$$0.45 - 0.5 \cdot 0.9 + k_1 = 0.0625 \rightarrow k_1 = 0.0625$$

.

The observer is therefore given by the equation

$$\hat{x}(k+1) = \begin{bmatrix} 0.5 & -0.0625 \\ 1.0 & 0.0 \end{bmatrix}\hat{x}(k) + \begin{bmatrix} 0 \\ 1 \end{bmatrix}u(k) + \begin{bmatrix} 0.0625 \\ 0.9 \end{bmatrix}y(k)$$

3. Consider the following task set.

Task	T_i	D_i	C_i
A	4	4	2
B	3	3	1
C	6	6	0.5

a. Using fixed-priority scheduling and rate-monotonic priority assignment, verify whether the the task set is schedulable or not. (1 p)

b. Using EDF scheduling, verify whether the task set is schedulable or not. (1 p)

Solution

a. The ordinary RMS scheduling condition gives that

$$\frac{2}{4} + \frac{1}{3} + \frac{0.5}{6} = 0.92 > 3(2^{1/3} - 1) = 0.78$$

Hence, using this we cannot tell whether the task set is schedulable or not. Using the hyperbolic condition leads to the same concludion since

$$(frac24 + 1)(\frac{1}{3} + 1)(\frac{0.5}{6} + 1) = 2.17 > 2.$$

Finally using response time analysis we have that

$$R_B^0 = 0, R_B^1 = C_B = 1 \leq D_B = 4$$

$$R_A^0 = 0, R_A^1 = C_A = 2,$$

$$R_A^2 = C_A + \left\lceil \frac{2}{T_B} \right\rceil C_B = 3,$$

$$R_A^3 = C_A + \left\lceil \frac{3}{T_B} \right\rceil C_B = 3 \leq D_A = 4$$

$$R_C^0 = 0, R_C^1 = C_C = 0.5,$$

$$R_C^2 = C_C + \left\lceil \frac{0.5}{T_A} \right\rceil C_A + \left\lceil \frac{0.5}{T_B} \right\rceil C_B = C_C + C_A + C_B = 3.5$$

$$R_C^3 = C_C + \left\lceil \frac{3.5}{T_A} \right\rceil C_A + \left\lceil \frac{3.5}{T_B} \right\rceil C_B = C_C + C_A + 2C_B = 4.5$$

$$R_C^4 = C_C + \left\lceil \frac{4.5}{T_A} \right\rceil C_A + \left\lceil \frac{4.5}{T_B} \right\rceil C_B = C_C + 2C_A + 2C_B = 6.5 > D_C = 6$$

Hence, the task set is not schedulable.

b. Under EDF the schedulability conditions is

$$\frac{2}{4} + \frac{1}{3} + \frac{0.5}{6} = 0.92 < 1$$

Hence, the task set is schedulable under EDF scheduling.

4. When implementing a controller, it is important to minimize the computational delay between the A-D conversion and the D-A conversion. The following equations describe a digital controller with Kalman filtering and state feedback:

$$\hat{x}(k) = (I - KC)\big(\Phi\hat{x}(k-1) + \Gamma u(k-1)\big) + Ky(k)$$
$$u(k) = -L\hat{x}(k)$$

In the general multi-input, multi-output (MIMO) case, if the process has r inputs, p outputs, and n states, then Φ is an $(n \times n)$ matrix, Γ is an $(n \times r)$ matrix, C is a $(p \times n)$ matrix, L is an $(r \times n)$ matrix, and K is an $(n \times p)$ matrix.

The controller should be implemented like this:

```
LOOP
ReadInputs;
CalculateOutputs;
WriteOutputs;
UpdateStates;
t := t + h;
WaitUntil(t);
END
```

a. What calculations must be performed in `CalculateOutputs` and what calculations can be postponed to `UpdateStates`? Write pseudo-code for both parts, and also point out things that can be pre-calculated before entering the control loop. (2 p)

b. Calculate the maximum number of floating-point operations (flops) carried out in each part, as functions of r, p, and n. An addition, a subtraction, or a multiplication of two scalars all count as one flop. Finally, specialize to the single-input, single-output (SISO) case with $r = 1$, $p = 1$, and $n = 4$. (2 p)

Solution

a. The `CalculateOutputs` part can be reduced to

```
u := u1 - LK*y;
```

where `u1` is the part of the control signal that can be computed in the previous sample. The factor `-L` can be precalculated outside the loop. The `UpdateStates` part can be written in different ways. The most straightforward implementation is perhaps

```
x := x1 + K*y;                (* Finish current state estimate *)
x1 := (I-KC)Phi*x + Gamma*u; (* Prepare next state estimate *)
u1 := -L*x1;                  (* Prepare next control signal *)
```

The factors `(I-KC)Phi` and `-L` can be precalculated. A more clever solution is to introduce the state variable $w(k) = \hat{x}(k) - Ky(k)$ and write the controller on general state-space form. The `UpdateStates` part can then written

```
w := (I-KC)(Phi-GammaL)*w + (I-KC)(Phi-GammaL)K*y;
u1 := -L*w;
```

where the factors `(I-KC)(Phi-GammaL)`, `(I-KC)(Phi-GammaL)K`, and `-L` can be precalculated.

b. A multiplication between two matrices of sizes $(a \times b)$ and $(b \times c)$ requires abc multiplications and $a(b-1)c$ additions, in total $a(2b-1)c$ flops. `CalculateOutputs`: `u1 - LK*y` requires $r + r(2p-1) = 2rp$ flops. In the special case $r = 1$, $p = 1$, $n = 4$, we get 2 flops. `UpdateStates`: In the first solution, `x1 + K*y` requires $n + n(2p-1)$ flops, `(I-KC)Phi*x + Gamma*u` requires $n + n(2n-1) + n(2r-1)$ flops, and `-L*x` requires $r(2n-1)$ flops. In total, we get $2n^2 + 2np + 4nr - n - r$ flops, and in the special case we get 51 flops.

In the second solution, `(I-KC)(Phi-GammaL)*w + (I-KC)(Phi-GammaL)K*y` requires $n + n(2n-1) + n(2p-1)$ flops, and `-L*w` requires $r(2n-1)$ flops. In total, we get $2n^2 + 2np + 2nr - n - r$ flops, and in the special case we get 43 flops.

5. Henry Hacker has implemented a real-time application with three critical common resources R1, R2 and R3 protected by mutual exclusion semaphores accessed by three different processes P1, P2 and P3. The critical sections in the three processes are accessed through the following statements:

P1	P2	P3
Wait(R1);	Wait(R2);	Wait(R3);
Wait(R2);	Wait(R3);	Wait(R1);
// Using R1 and R2	// Using R2 and R3	//Using R1 and R3
Signal(R2);	Signal(R3);	Signal(R1);
Signal(R1);	Signal(R2);	Signal(R3);

Do you see a problem with this solution? If so, explain why and suggest an implementation avoiding the problem. (2 p)

Solution

The problem is that there is a circular wait-chain. If all processes execute the first Wait-statement, they all have acquired a resource. In the next statement, they all try to acquire another resource which is already held by one of the other processes. Deadlock will occur. This may be solved by using hierarchical resource allocation and rewriting P3 to:

```
Wait(R1);
Wait(R3);
// Using R1 and R3
Signal(R3);
Signal(R1);
```

6. An industrial process with a long time delay is described by the continuous-time transfer function

$$G(s) = \frac{1}{(1+10s)^2} e^{-30s}$$

Discretize the process using ZOH and the sampling interval $h = 1$ and write down the resulting transfer function $H(z)$. What is the order of the resulting discrete-time system? (2 p)

Solution

Since the delay $\tau = 30$ is an integer multiple of the sampling interval $h = 1$, it will translate into a backward shift of $\tau/h = 30$ samples in the discrete-time system. Sampling the non-delayed part of the system using Table 3 in IFAC PB, we obtain

$$H'(z) = \frac{1.1(1-e^{-0.1})z + e^{-0.1}(e^{-0.1} - 0.9)}{(z-e^{-0.1})^2} = \frac{0.004679z + 0.004377}{z^2 - 1.81z + 0.8187}$$

Including the delay we have

$$H(z) = \frac{0.004679z + 0.004377}{(z^2 - 1.81z + 0.8187)z^{30}}$$

The system order is given by the order of the denominator polynomial: 32.

7.

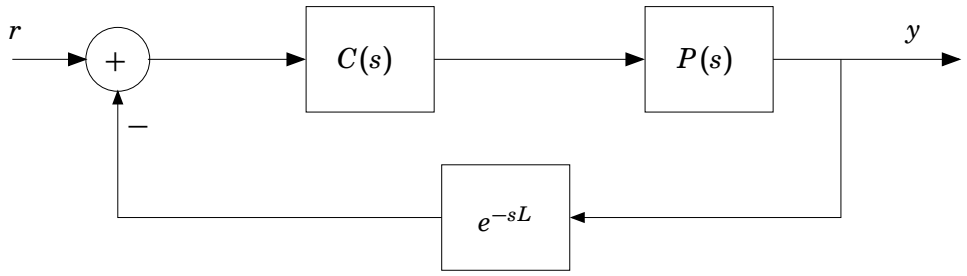

Figure 1 Closed loop system in Problem 7, where the controller $C(s)$ is one of $C_1(s)$, $C_2(s)$ or $C_3(s)$.

A model for the position of a robot arm used for handling of radioactive materials is given by the transfer function

$$P(s) = \frac{1}{s^2}.$$

The process is to be controlled using a PD-controller of the form

$$C(s) = K\left(1 + \frac{sT_d}{1 + sT_d/N}\right)$$

as shown in Figure 1, where the measurements of the position y are obtained from some type of position sensor. There are three types of position sensors available to choose between. Each of these sensors has different noise characteristics, and therefore the parameter N in the controller needs to be set differently depending of the choice of sensor. In addition, each sensor introduces a different time delay, due to the different signal processing algorithms needed to obtain the position measurement. The available sensors and the relevant data are:

1. Laser range sensor, used with controller $C_1(s)$ with $N = 100$, time delay $L = 5$ ms, price \$10.000.
2. High-resolution Digital CCD Camera, used with controller $C_2(s)$ with $N = 8$, time delay $L = 30$ ms, price \$2.000.
3. Low-resolution digital CCD Camera, used with controller $C_3(s)$ with $N = 2$, time delay $L = 15$ ms, price \$500.

The Bode diagrams of the open-loop systems $C_1(s)P(s)$ (solid line), $C_2(s)P(s)$ (dashed line) and $C_3(s)P(s)$ (dotted line) *without time delay* can be seen in Figure 2.

a. Based on this information, and if a low cost is desired, which solution/sensor would you recommend? (2 p)

b. Discretize the controller $C(s)$ using the sampling time h and the backward difference approximation. Show how the control signal $u(k)$ at sample k can be calculated from the control errors ($r(k) - y(k)$, $r(k-1) - y(k-1)$,...) and previous values of the control signal and controller states. Determine which part of the control signal $u(k)$ can be pre-computed at sample $(k-1)$. (1.5 p)

Solution

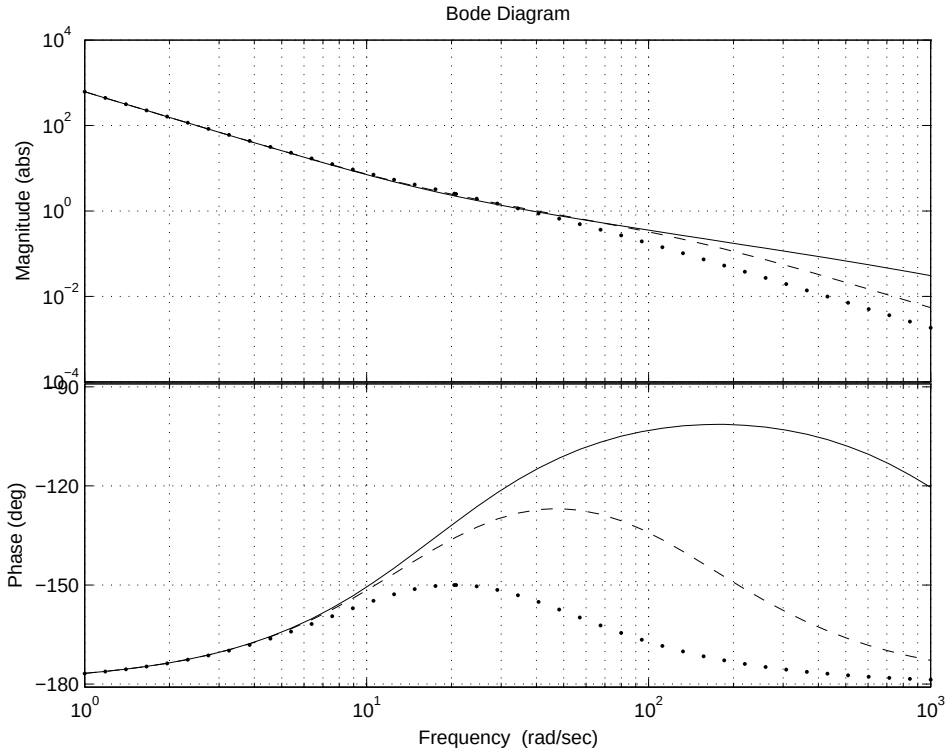

Figure 2 Open-loop Bode diagrams $C_1(s)P(s)$ (solid line), $C_2(s)P(s)$ (dashed line) and $C_3(s)P(s)$ (dotted line) in Problem 7.

a. From the Bode diagram we see that the phase margins are 65° for $C_1(s)$, 50° for $C_2(s)$ and 25° for $C_3(s)$at the crossover frequency $\omega_c \approx 40$ rad/s. Therefore, the corresponding delay margins are given by $L_1^{max} = 65\pi/180/40 = 28$ ms, $L_2^{max} = 55\pi/180/40 = 22$ ms, and $L_3^{max} = 25\pi/180/40 = 11$ ms. We see that time delay for system 2 and 3 are longer than the corresponding delay margins, and the systems will be unstable. Therefore, only sensor 1 can be recommended.

b. From the block diagram we see that

$$U(s) = K(1 + \frac{sT_d}{1 + sT_d/N})E(s)$$

with

$$E(s) = R(s) - Y(s)$$

or equivalently $e(t) = r(t) - y(t)$. With backward Euler we replace s with $\frac{q-1}{qh}$ and get

$$u(k) = Ke(k) + K\frac{T_d\frac{q-1}{qh}}{1 + \frac{T_d(q-1)}{qhN}}e(k) = Ke(k) + D(k)$$

with $D(k)$ given by

$$D(k) = K\frac{T_d\frac{q-1}{qh}}{1 + \frac{T_d(q-1)}{qhN}}e(k) = K\frac{NT_d(q-1)}{qNh + T_d(q-1)}e(k) = \frac{KNT_d(q-1)}{q(Nh + T_d) - T_d)}e(k)$$

which can be written as

$$D(k+1) = \frac{T_d}{Nh+T_d}D(k) + \frac{KNT_d}{Nh+T_d}(e(k+1) - e(k)).$$

The control signal is therefore given by $u(k) = Ke(k) + D(k)$ or

$$\begin{aligned} u(k) &= Ke(k) + \frac{T_d}{Nh+T_d}D(k-1) + \frac{KNT_d}{Nh+T_d}(e(k) - e(k-1)) = \\ &= (K + \frac{KNT_d}{Nh+T_d})e(k) - \underbrace{\frac{KNT_d}{Nh+T_d}e(k-1) + \frac{T_d}{Nh+T_d}D(k-1)}_{\text{can be pre-computed at time k-1}} \end{aligned}$$

8. Consider a simple chemical process that combines two reagents, R1 and R2, to produce the final result. It works by first pouring enough of one reagent into a container to reach a particular level, then pouring enough of the second reagent until a second level is reached (while mixing both reagents), and then emptying the container.

Figure 3 shows the system.

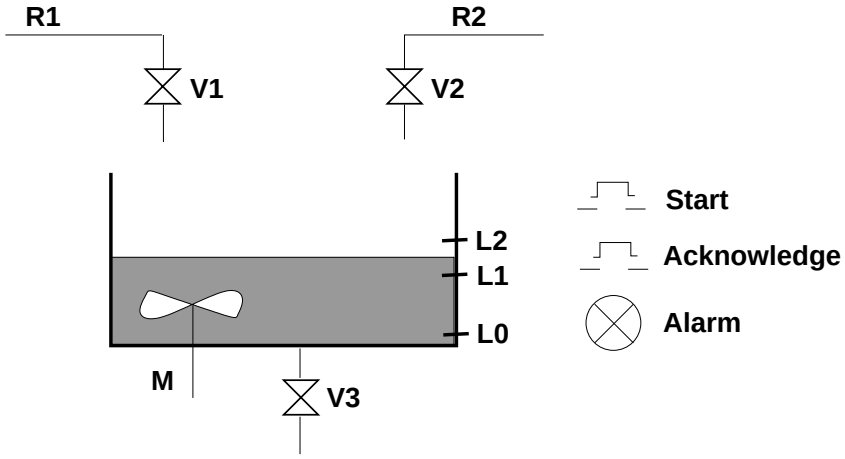

Figure 3 Simple chemical process.

Develop a Grafcet controller that guarantees the following desired sequence of operations:

1. When the start button is pressed, V1 should be opened until level L1 is reached.
2. When L1 is reached, the mixer should start mixing and simultaneously V2 should be opened.
3. When L2 is reached, the mixer should stop, and V3 should be opened until the tank's level goes below L0.
4. If, after 10 minutes, the level of the tank is not below L0, an alarm is started. The "acknowledge" button stops the alarm and allows restarting of the control sequence.

You may use the notation <stepname>.x (true when the step is active) and <stepname>.s (the number of seconds that has elapsed since the step last was activated) in your transitions. In order to get full points, the alarm management should be performed in a Grafcet that is separate from the Grafcet for the filling and emptying of the tank, and no extra variables/signals may be used. (3 p)

Solution

The solution is given in Fig 4.

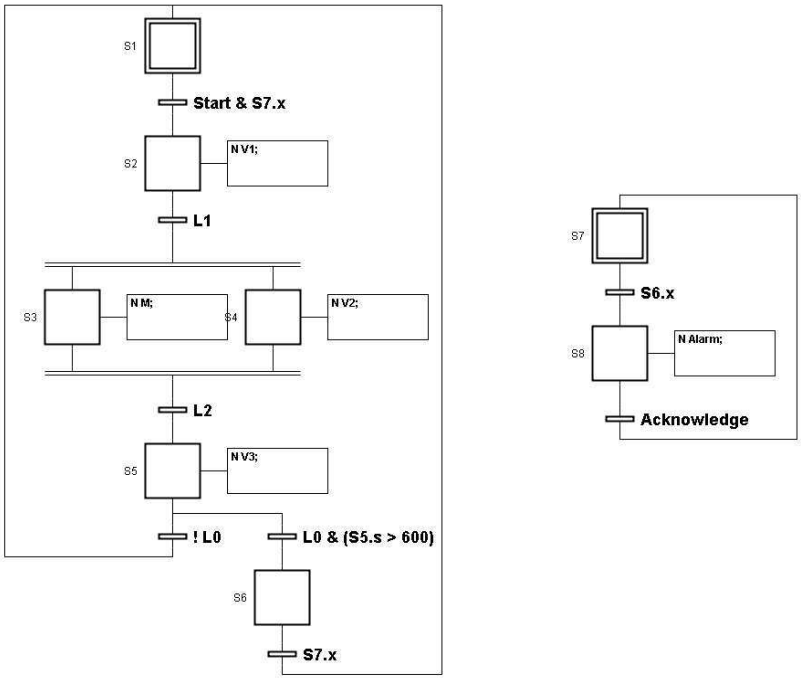

Figure 4 Grafcet for simple chemical process.

9. Consider the following two Java classes:

```
public class MyObject {

public int a = 2;
public static int b = 1;

}

public class MyClass {

public static void myMethod(int a, MyObject obj) {
a = 0;
obj.a = 3;
obj.b = 9;
}
```

```
public static void main(String[] args) {
int a = 3;
MyObject ref1 = new MyObject();
MyObject ref2 = new MyObject();
a = 7;
myMethod(a,ref1);
System.out.println("Output 1 = " + a);
System.out.println("Output 2 = " + ref1.a);
System.out.println("Output 3 = " + ref2.b);
}
}
```

a. What will printed on the screen when the class `MyClass` is executed, i.e., `> java MyClass`? (1 p)

b. Explain why this is the case. (2 p)

Solution

a. The output will be

```
Output 1 = 7
Output 2 = 3
Output 3 = 9
```

b. `a` is a local variable which is assigned the value `7` in the main method. The fact that `a` is passed as an argument to `myMethod` will not change its value, since Java uses call-by-value when simple data types are used as method arguments. `ref1` is a reference to a `MyClass` object, which is then passed as an argument to the method `myMethod`. Since Java uses call-by-reference when passing objects as arguments to methods the assignment `obj.a = 3` will be performed on the object referenced by `ref1` (actually Java uses call-by value also in this case since it is a reference that is the argument to the method). Since `b` is declared as `static` all instances of `MyClass` will share this attribute. Hence, the assignment `obj.b = 9` in `myMethod` will effect also the object referenced by `ref2`.