

These suggested solutions are compiled by Kisimi Kayleemasa. Efforts were made by Kisimi to carryout test driven development to ensure the correctness of the codes in these solutions. However, since he is just a student just about to complete his first semester, no 100% guarantee can be given about the correctness of the codes. You are therefore free to make comments and suggestions which can lead to the improvement of the codes.

The good part is all the codes here are copied and pasted from a test driven project Kisimi implemented and the programme functions as desired. You can have the full package if you need. I will find a place to publish it to all. The package also includes the JUNIT test classes.

Oppgave 1. (15%) (3 deloppgave)

(3%) Deloppgave 1 a
Hva vil dette programmet skrive ut når det blir utført?
Svar: 64

```
public static void Oppgavela(){
    int[] tab = {2, 4, 8, 8, 16, 32, 64};
    for(int i = tab.length-1; i > 0; i--){
        tab[i-1] = tab[i];
    }
    System.out.println(tab[0]);
}
```

Explanation of answer:

The start value of "i" in the for loop is 6. Miraculously, the answer is 64. But let me try to explain what happens here:

```
tab[6-1] = tab[6]
```

Note that tab[6] is 64 and this value will be assigned to tab[5] implying that, in the next run, the value to be assigned will also be 64 because the next is tab[5] which is 64.

```
tab[5-1] = tab[5]
```

You see it now. It means again that the value of the next candidate which is tab[4] is 64. This confirms that, on the right hand side, the value will always be 64.

The purpose of examination is not to trick students but the java programming language does that. Well, we have to live with that. Examination is not the best way to test students. We are not compilers. The trend nowadays is test driven programming which means, programmers need to test as they implement to see some of these hidden facts about the programming language.

(3%) Deloppgave 1 b
Hva vil dette programmet skrive ut når det blir utført?
Svar:

2
0
3

```
public class Oppgavelb{
    public static void main(String[] args){
        int i = 2;
        int j = 3;
        System.out.println(beregn(1, 1));
        System.out.println(beregn(i, j));
        System.out.println(beregn(j, i));
    }

    public static int beregn(int i, int j){
        i %= 2;
        j %= 3;
        return i + j;
    }
}
```

Explanation of answer:

The operator % means the rest when a division is carried out. When this function is called as beregn(1, 1), it will do the following:

```
i = 1 % 2 = 1 ; j = 1 % 3 = 1 ; i + j = 1 + 1 = 2
2 is therefore printed out in the first call
```

When it is called as beregn(i, j) when i = 2 and j = 3 (declared in main method - see code in exams paper), see main method up, we get the following:

```
i = 2 % 2 = 0 ; j = 3 % 3 = 0 ; i + j = 0 + 0 = 0
0 is therefore printed out in the second call.
```

When it is called as beregn(j, i) when i = 3 and j = 2, see main method up, we get the following:

```
i = 3 % 2 = 1 ; j = 2 % 3 = 2 ; j + i = 2 + 1 = 3
3 is therefore printed out in the third call
```

Deloppgave 1c

Hva vil dette programmet skrive ut når det blir utført?

Svar:

2.5

3.0

3.0

```
Public class Oppgavelc{
    public static void main(String[] args){
        Dings s = new Dings();
        s.leggSammen(2.0);
        s.leggSammen(3.0);
        System.out.println(s.gjsnitt());
        Dings t = s;
        t.leggSammen(4.0);
        System.out.println(t.gjsnitt());
        System.out.println(s.gjsnitt());
    }

    public class Dings {
        private int n;
        private double s;
        public void leggSammen(double t){
            s += t;
            n++;
        }
        public double gjsnitt() { return s/n; }
    }
}
```

Explanation of answer:

We first create an instance of Dings called s.

The first call of the method leggSammen(t) will then do the following (see code of the method in the class Dings):

```
s = s + 2.0 = 0.0 + 2.0 = 2.0 because at the start s is 0.0
n++ ==> n is now 1 after the execution of the method
```

The second call leggSammen(t) does the following

```
s = s + 3.0 = 2.0 + 3.0 because now s was already 2.0 during the first call
n++ ==> n is now 2 after the second call
```

```
When s.gjsnitt() is called, it will return the average which is simply
s/n = 5.0 / 2.0 = 2.5
```

Now, another instance of Dings is created called t and on creation, it points directly to the address of the previous instance s. This means, any change done to t will equally affect s because both point to the same address. So when t.leggSammen(4.0) is called, we have,

```
s = s + 4.0 = 5.0 + 4.0 = 9.0
```

because s is 5 after the second call of the method leggSammen()

```
n++ ==> n is now 3
```

By calling gjsnitt() as shown below,

```
System.out.println(t.gjsnitt());
System.out.println(s.gjsnitt());
```

we will have the same output for both t and s as shown below:

```
9.0/3.0 = 3.0
```

```
9.0/3.0 = 3.0
```

```

(3%) Deloppgave 3a
    public Pasientjournal(int pasientID) {
        this.pasientID = pasientID;
        this.behandlinger = new Behandling[ANTALL_DAGER];
    }

(3%) Deloppgave 3b
    public static boolean lovligDagnr(int _dagnr){
        if(_dagnr>=0 && _dagnr<7){
            return true;
        }

        return false;
    }

(3%) Deloppgave 3c
    public static String hentDagNavn(int _dagnr){
        if(!(lovligDagnr(_dagnr))){
            return null;
        }

        return ukedager[_dagnr];
    }

(3%) Deloppgave 3d
    public boolean innsettBehandlingForGittDag(int _dagnr, int tid, Helsepersonell
        person, String merknad){

        boolean status = false;

        if(!(lovligDagnr(_dagnr))){
            return status;
        }

        if(behandlinger[_dagnr] == null){
            Behandling behandling = new Behandling(tid, person, merknad);
            behandlinger[_dagnr] = behandling;
            antallB++;
            status = true;
        } else {
            System.out.println("Patient has already received treatment this day of
                " + hentDagNavn(_dagnr));
            return status;
        }
        return status;
    }

(3%) Deloppgave 3e
    public Behandling hentBehandlingForGittDag(int _dagnr){
        if(!(lovligDagnr(_dagnr))){
            return null;
        }

        return behandlinger[_dagnr];
    }

(3%) Deloppgave 3f
    public boolean equals(Object obj){

        if(this == obj) return true;

        Pasientjournal pjournal = null;
        if(obj instanceof Pasientjournal)
            pjournal = (Pasientjournal) obj;

        return this.hentID() == pjournal.hentID();
    }

```

(3%) Deloppgave 3g

```
public int compareTo(Object obj){
    Random r = new Random();
    int negativeX = 0 - r.nextInt(1000);
    int positiveY = 1 + r.nextInt(1000);
    if(this == obj)
        return 0;

    Pasientjournal pjournal = null;
    if(obj instanceof Pasientjournal)
        pjournal = (Pasientjournal) obj;

    if(this.hentID()==pjournals.hentID()){
        return 0;
    } else if(this.hentID()>pjournal.hentID()){
        return positiveY;
    } else {
        return negativeX;
    }
}
```

(4%) Deloppgave 3h

```
public String formaterBehandling(){

    String pos = "";
    String bInfo = "";
    String dag = null;
    for(int i=0; i<ANTALL_DAGER; i++){
        if(behandlinger[i] != null){
            pos = behandlinger[i].hentHelsemedarbeider().hentStilling()
                .toString();

            switch(i){
                case 0 :
                    dag = "mandag";
                    break;
                case 1 :
                    dag = "tirsdag";
                    break;
                case 2 :
                    dag = "onsdag";
                    break;
                case 3 :
                    dag = "torsdag";
                    break;
                case 4 :
                    dag = "fredag";
                    break;
                case 5 :
                    dag = "lørdag";
                    break;
                case 6 :
                    dag = "søndag";
                    break;
                default:
                    assert false;
            }

            bInfo += String.format("%10s%5s %s(%-1s)\t%-1s%n", dag,
                behandlinger[i].hentTid(),
                behandlinger[i].hentHelsemedarbeider().hentNavn(),
                pos, behandlinger[i].hentMerknad());
        }
    }
    return bInfo;
}
```

(4%) Deloppgave 4a

```
public Pasientjournal finnPJ(int pID){
    Pasientjournal p = null;
    for(int i=0; i<antallPJ; i++){
        if(pasientjournaler[i].hentID() == pID){
            p = pasientjournaler[i];
        }
    }
    return p;
}
```

(4%) Deloppgave 4b

```
public boolean registrerPJ(int pID){
    boolean status = false;
    Pasientjournal pj = finnPJ(pID);
    if(pj != null){
        System.out.println("Pasientjournalen finnes fra før!");
        return status;
    }

    if(antallPJ < pasientjournaler.length){
        pj = new Pasientjournal(pID);
        pasientjournaler[antallPJ++] = pj;
        status = true;
    }

    return status;
}
```

(4%) Deloppgave 4c

```
public boolean innsettBehandlingForGittPasientForGittDag(int pID, int dagnr, int tid,
Helsepersonell person, String merknad){

    boolean status = false;
    if(!(Pasientjournal.lovligDagnr(dagnr)))
        return status;

    Pasientjournal nyPj = finnPJ(pID);

    if(nyPj == null){
        registrerPJ(pID);
        return finnPJ(pID).innsettBehandlingForGittDag(dagnr, tid, person,
merknad);
    }

    return nyPj.innsettBehandlingForGittDag(dagnr, tid, person, merknad);
}
```

(4%) Deloppgave 4d

```
public Behandling hentBehandlingForGittPasientForGittDag(int pID, int dagnr){

    Behandling behandling = null;

    if(!(Pasientjournal.lovligDagnr(dagnr))){
        System.out.println("Ulovlig dagniummmmer");
        return null;
    }

    Pasientjournal nyPj = finnPJ(pID);

    if(nyPj == null)
        return null;

    return behandling = nyPj.hentBehandlingForGittDag(dagnr);
}
```

(12%) Deloppgave 4e

```
private boolean innsettPost(String post) {

    int feltSlutt1 = post.indexOf(FELT_SLUTT_TEGN);
    int feltSlutt2 = post.indexOf(FELT_SLUTT_TEGN, feltSlutt1+1);
    int feltSlutt3 = post.indexOf(FELT_SLUTT_TEGN, feltSlutt2+1);
    int feltSlutt4 = post.indexOf(FELT_SLUTT_TEGN, feltSlutt3+1);
    int feltSlutt5 = post.indexOf(FELT_SLUTT_TEGN, feltSlutt4+1);
    try {
        //Parsing integer strings here - throws exception if successful
        int pID = Integer.parseInt(post.substring(0, feltSlutt1));
        int dagnr = Integer.parseInt(post.substring(feltSlutt1 + 1,
            feltSlutt2));
        int bTid = Integer.parseInt(post.substring(feltSlutt2 + 1,
            feltSlutt3));

        String medarbeider = post.substring(feltSlutt3 + 1, feltSlutt4);
        String mStilling = post.substring(feltSlutt4 + 1, feltSlutt5);
        String merknad = post.substring(feltSlutt5 + 1);

        // Kontroller danummer:
        if (Pasientjournal.lovligDagnr(dagnr) == false) {
            System.out.println("Ulovlig dagnummer i post: " + post);
            return false;
        }

        Helsepersonell hP = new Helsepersonell(medarbeider,
            Stilling.valueOf(mStilling));

        Pasientjournal pj = finnPJ(pID);
        boolean ok = true;
        if (pj != null){
            pj = finnPJ(pID);
            pj.innsettBehandlingForGittDag(bTid, pID, hP, merknad);
        }
        else {
            ok = registrerPJ(pID);
            pj = finnPJ(pID);
        }

        if (ok){
            ok = pj.innsettBehandlingForGittDag(dagnr, bTid, hP, merknad);
        }

        if (!ok){
            System.out.println("Posten ble ikke lagt inn: " + post);
            return false;
        }

        return ok;
    } catch (NumberFormatException formateringsUnntak) {
        System.out.println("Feil ved lesing av heltalsverdier fra post:
            " + post);
        return false;
    }
}
```

(8%) Deloppgave 4f

```
public void tidForbruk(){
    int tTidPleier = 0;
    int tTidLege = 0;

    for(int i=0; i<antallPJ; i++){
        for(int dnr=0; dnr<Pasientjournal.ANTALL_DAGER; dnr++){
            if(pasientjournaler[i].hentBehandlingForGittDag(dnr) != null){
                if(pasientjournaler[i].hentBehandlingForGittDag(dnr).
                    hentHelsemedarbeider().hentStilling().
                    toString().equals("LEGE")){

                    tTidLege += pasientjournaler[i].
                        hentBehandlingForGittDag(dnr).hentTid();
                }

                if(pasientjournaler[i].hentBehandlingForGittDag(dnr).
                    hentHelsemedarbeider().hentStilling().
                    toString().equals("PLEIER")){

                    tTidPleier += pasientjournaler[i].
                        hentBehandlingForGittDag(dnr).hentTid();
                }
            }
        }
    }

    int lTimer = (tTidLege - (tTidLege % 60)) / 60;
    int lMinuter = tTidLege % 60;
    int pTimer = (tTidPleier - (tTidPleier % 60)) / 60;
    int pMinuter = tTidPleier % 60;

    System.out.println("TIDSFORBRUK:\nTid til behandling (leger): " + lTimer +"t.
        " + lMinuter + " min. ");
    System.out.println("Tid til behandling (pleiere): " + pTimer +"t. " + pMinuter
        + " min.");
}
```

(15%) Deloppgave 4g

```
public void skrivUkerapport(){
    int pID;
    int totalWeeklyNumberOfTreatmentsPerPatient = 0;
    int totalWeeklyNumberOfTreatmentsForAllPatients = 0;
    int totalTreatmentTimePerPatient = 0;
    int totalTreatmentTimeForAllPatients = 0;
    StringBuilder isStar = new StringBuilder("");
    final int ukeDager[] = new int[7];

    System.out.println("UKERAPPORT:");
    System.out.printf("%-20s%-8s%-8s%-8s%-8s%-8s%-8s%-8s%-8s%-8s\n",
        "Pasient nr.", "M", "T", "O", "T", "F", "L", "S", "Tid",
        "Antall");
    for(int i=0; i<antallPJ;i++){
        pID = pasientjournaler[i].hentID();
        isStar.append(String.format("%-20s",pID));
        for(int j=0; j<Pasientjournal.ANTALL_DAGER; j++){
            if(pasientjournaler[i].hentBehandlingForGittDag(j) != null)
                ukeDager[j]++;
            isStar.append(String.format("%-8s", myStarCamilla(i, j)));
            if(pasientjournaler[i].hentBehandlingForGittDag(j) != null){
                totalTreatmentTimePerPatient +=
                    pasientjournaler[i].hentBehandlingForGittDag(j).
                    hentTid();

                totalWeeklyNumberOfTreatmentsPerPatient++;
            }
            pID = ' ';
        }

        isStar.append(String.format("%-8d%-8d", totalTreatmentTimePerPatient,
            totalWeeklyNumberOfTreatmentsPerPatient));

        if(i < antallPJ-1)
            isStar.append(String.format("\n"));

        totalTreatmentTimeForAllPatients += totalTreatmentTimePerPatient;
        totalWeeklyNumberOfTreatmentsForAllPatients +=
            totalWeeklyNumberOfTreatmentsPerPatient;
        totalTreatmentTimePerPatient = 0;
        totalWeeklyNumberOfTreatmentsPerPatient = 0;
    }
    System.out.println(isStar);
    System.out.printf("%-20s%-8s%-8s%-8s%-8s%-8s%-8s%-8d%-8d\n",
        "TOTAL:", ukeDager[0], ukeDager[1], ukeDager[2], ukeDager[3],
        ukeDager[4], ukeDager[5], ukeDager[6],
        totalTreatmentTimeForAllPatients,
        totalWeeklyNumberOfTreatmentsForAllPatients);
}
```

This is just a helper method used in Deloppgave 4g

```
private char myStarCamilla(int i, int j) {
    char isStar;
    if(pasientjournaler[i].hentBehandlingForGittDag(j) != null){
        isStar = '*';
    } else
        isStar = ' ';
    return isStar;
}
```

(4%) Deloppgave 4h

```
public void skrivPasientJournaler(){
    System.out.println("ALLE PASIENTJOURNALER:");
    Hjelpklasse.sortVedUtvalg(pasientjournaler, antallPJ);
    for(int i=0; i<antallPJ; i++){
        System.out.println(pasientjournaler[i].toString());
    }
}
```