

i Prófleiðbeiningar

Einu leyfilegu hjálpargögnin eru **eitt A4 blað (handskrifað báðum megin)**, sem nemandinn hefur búið til.

Ef þú **sérð ekki** hluta prófsins, t.d. hlekki eða myndir, gefðu þig fram við **prófvörð**

Ef **galli á prófinu kemur í ljós**, verður tekið tillit til þess við yfirferð prófsins.

Ef nemandi er óviss um hvort skilningur hans á forritunarspurningu er réttur, getur hann tilgreint hver sá skilningur er í svarreit í opnum spurningum eða sent póst til kennara eftir prófið

Kynnið ykkur **viðhengi** sem eru með prófinu.

Farið verður **nafnlaust** yfir prófið þannig að þið skuluð ekki skrifa nafnið ykkar í lausnina.

i Leiðbeiningar krossar

Svarið eftirfarandi krossaspurningum með því að merkja við eitt svar í hverri spurningu. Ef ykkur finnst fleiri en eitt svar koma til greina skuluð þið velja svarið sem ykkur finnst „réttast“. Ekki er dregið niður fyrir rangt svar.

1 Segð

Hvert er gildið (e. value) á eftirfarandi segð (e. expression) ef b hefur gildið 2, a hefur gildið 2 og c hefur gildið 1

$b++ * b - 4.0 * a * c$

Veljið eitt af eftirtöldu:

☐ -2

☐ -4.0

☐ -2.0

☐ -4

☐ 2

Maximum marks: 4

2 Tvo

Skoðið eftirfarandi forritsbút. Hvað eru prentaðar margar línur á staðalúttak í eftirfarandi keyrslu?

```
%java Tvo 4
```

```
public class Tvo {  
    public static int thridja(int i) {  
        i = i * i * i;  
        return i;  
    }  
    public static void main(String[] args) {  
        int n = Integer.parseInt(args[0]);  
        for (int i = 1; i < n; i++) {  
            System.out.println(i + " " + thridja(i));  
        }  
    }  
}
```

Veljið eitt af eftirtöldu:

- ☐ 1
- ☐ 2
- ☐ 4
- ☐ 5
- ☐ 3

Maximum marks: 4

3 Sniðið úttak

Ef **verd** og **afsl** eru **double** breytur með gildin 452.5 og 15.0 hver eftirfarandi **printf** setninga prentar út eftirfarandi texta:

Verð: 452.50, afsláttur: 15%

Veljið eitt af eftirtöldu:

- ☐ System.out.printf(Locale.US,"Verð: %.2f, afsláttur: %.0f%% \n"+ verd+afsl);
- ☐ System.out.printf(Locale.US,"Verð: %.2f, afsláttur: %.0f%% \n", verd, afsl);
- ☐ System.out.printf(Locale.US,"Verð: %.2, afsláttur: %.0f%% \n", verd, afsl);
- ☐ System.out.printf(Locale.US,"Verð: %.2f, afsláttur: %.0f%% \n", verd+afsl);
- ☐ System.out.printf(Locale.US,"Verð: %.2f, afsláttur: %.0f \n", verd, afsl);

Maximum marks: 4

4 Endurkvæmt deilanlegt

Skoðið eftirfarandi forrit. Hvað er prentað út í eftirfarandi keyrslu ?

%java Fjogur 4

```
public class Fjogur {
    public static int deilanlegt(int n, int[] a) {
        if (n == 0)
            return 0;
        else
            return (a[n] % n == 0 ? 1 : 0) + deilanlegt(n - 1, a);
    }
    public static void main(String[] args) {
        int[] tala = { 0, 1, 3, 6, 8 };
        System.out.println(deilanlegt(Integer.parseInt(args[0]), tala));
    }
}
```

Veldu eitt af eftirtöldu:

- ☐ 3
- ☐ 0
- ☐ 4
- ☐ 2
- ☐ ArithmeticException

Maximum marks: 4

main forritið les inn **eina heiltölu** (á bilinu 1 til 10 báðar meðtaldar) af **staðalinntaki** sem er gæði kartaflna og prentar út **flokkinn** á staðalúttak. Forritið flokkar kartöflur í fjóra flokka eftir gæðum, flokka 0, 6, 9 og 10. Ef kartöflur eru af gæðum 0 þá fara kartöflurnar í flokk 0, ef þær eru á bilinu 1 til og með 6 fara þær í flokk 6, ef þær eru af gæðum hærri en 6 og minni en 10 fara þær í flokk 9. Ef þær eru af gæðum 10 þá fara þær í flokk 10.

Notið **System.out.println** til að prenta út flokkinn.

Prófið forritið (Test code) og skoðið vel útkomuna úr prófanatilvikunum (test cases). Athugið að prófanatilvikin eru ekki tæmandi.

5 Flokka

Forritið hér

Test case #	Input	Expected output
1	5	6
2	0	0

```
import java.util.Locale;
import java.util.Scanner;

class Kartofflur {

    public static void main(String[] args) {
        Scanner s = new Scanner(System.in, "UTF-8");
        s.useLocale(Locale.US);
        // forritið hér
    }
}
```

Test code

Maximum marks: 8

i Leiðbeiningar forritunarspurningar

Í forritunardæmum 6-11 er gefinn **java ritill** til að forrita í. Ritillinn er með setninganúmerum og sýnir java highlights með lit. Ritillinn hjálpar ykkur með réttan inndrátt á línunum **ef þið notið slaufusviga**. Ritillinn athugar ekki hvort forritið er setningarlega (e. syntax) rétt og hann þýðir hvorki né keyrir forritið.

Notið **bestu venjur** við forritun. Notið aðeins þá klasa sem farið hefur verið yfir í námsefninu.

Þegar gefin er byrjun á forriti eða beinagrind getið þið afritað hana í ritilinn. Ef gefin eru leiðbeinandi línukomment farið eftir þeim leiðbeiningum og eyðið þeim ekki.

Þið þurfið **ekki að lýsa forritinu** með því að bæta við haus eða javaDoc nema sé beðið um það sérstaklega

Þið þurfið **ekki að bæta við import** setningum efst í forritið ykkar

Eftirfarandi forrit tekur inn **heiltölu af skipanalínu** og **prentar út árstíðina**. Dæmi um keyrslu er:

```
%java Arstid 3
Vor
```

Endurskrifið (Refactor) forritið þannig að það hafi nákvæmlega **sömu virkni** (e. functionality) (þ.e. inntak og úttak þarf að vera það sama) en sé **einfaldara** og **fylgi bestu venjum** í forritun.

```
---
public class Arstid {
    public static void main(String[] args) {
        int a = Integer.parseInt(args[0]);
        String Arstid="Ólöglegt";
        switch (a) {
            case 1:
                Arstid = "Sumar";
                System.out.print(Arstid);
                break;
            case 2:
                Arstid = "Vetur";
                System.out.print(Arstid);
                break;
            case 3:
                Arstid = "Vor";
                System.out.print(Arstid);
                break;
            case 4:
                Arstid = "Haust";
                System.out.print (Arstid);
        }
        System.out.println();
    }
}
```


6 Arstid - Refactoring - Java

Forritið hér

1

Maximum marks: 8

Skrifið aðalforrit í klasanum **Fiskur** sem les inn af staðalinntaki ótilgreindan fjölda mælinga af tveimur kommutölum thyngd1 og thyngd2 og tveimur heiltölum timi1 og timi2:

thyngd1 thyngd2 timi1 timi2

Þyngdirnar eru þyngd fisks á tíma eitt og tíma tvö. Þannig má sjá hvað fiskur hefur vaxið mikið í prósentum með formúlunni:

$$vöxtur = \frac{thyngd2 - thyngd1}{timi2 - timi1} * 100$$

Forritið reiknar út meðaltalið af vexti allra mælinga. Mælingar sem hafa timi2 <= timi1 á ekki að taka með.

Úttakið gæti verið svona fyrir eftirfarandi inntak

```
%java Fiskur
```

```
1.0 3.0 1 5
```

```
1.0 3.0 1 1
```

```
^D
```

Meðaltal vaxtar er 50.0

og hér er engin færsla tekin með og meðaltalið því skilgreint 0.0:

```
%java Fiskur
```

```
0.5 1.0 1 1
```

```
^D
```

Meðaltal vaxtar er 0.0

Afritið eftirfarandi í ritil:

```
public class Fiskur {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in, StandardCharsets.UTF_8);
        s.useLocale(Locale.US);
        // forritið hér ***

        // og hingað ***
    }
}
```

7 Fiskur - Java

Forritið hér

1	
---	--

Maximum marks: 8

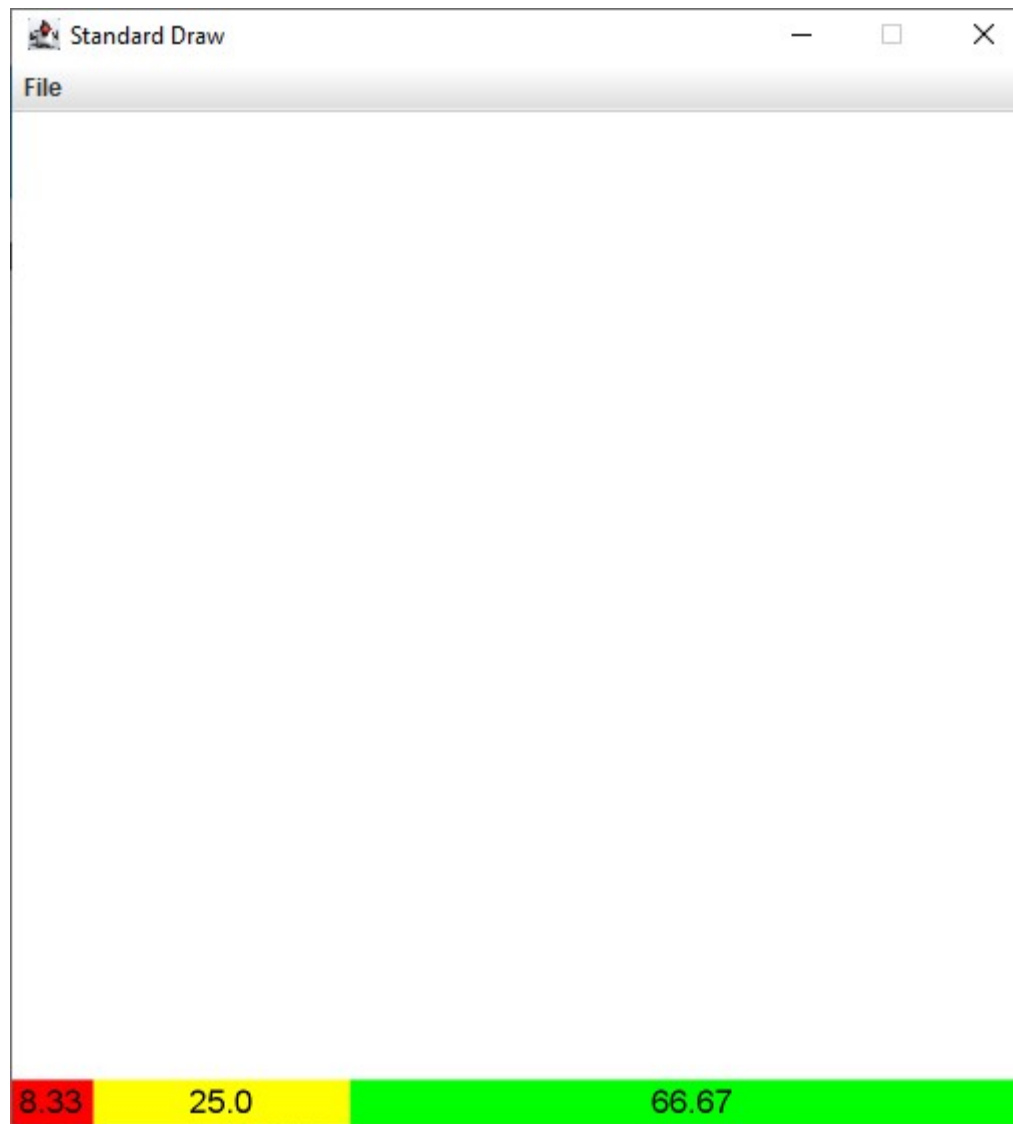
Skrifið aðferðina **teikna** sem hefur viðfangsbreytu (parameter) fylki af kommutölum sem samanlagðar hafa gildið 100.0. Aðferðin teiknar lárétt staflað súlurit þar sem fyrir hverja tölu er teiknaður hluti af súlu. Til einföldunar er aðeins teiknuð ein súla.

Litirnir ráðast af tölunni og eru ákvarðaðir í gefinni aðferð sem heitir **litur** og sem hefur parameter **kommutölu** og skilar **Color** gildi. Þið getið reiknað með að **litur** aðferðin sé public og static og gefin í klasanum **Litir**. Þið þurfið ekki að forrita hana. Mynsturfar (e. signature) er ekki gefið en þið eigið að geta séð hvert það er út frá ofangreindum upplýsingum.

Setjið skalann á X og Y ásnum hæfilegan. Á myndinni er X skalinn 100.0 og Y skalinn er sá sami. Stærðin á canvas-inu er sjálfgefin og þarf ekki að breyta.

Eftirfarandi aðferðir úr **StdDraw** klasanum gætu gagnast ykkur: `setPenColor`, `filledRectangle`, `text`, `setXscale`, `setYscale`, `setScale`. **StdDraw** klasann er að finna undir Resources.

Þegar forritið er keyrt birtist eftirfarandi mynd:



Afritið í ritilinn

```
public class StoplaritLarett {  
    // Forritið aðferðina teikna ***  
  
    // og hingað ***  
    public static void main(String[] args) {  
        double[] tolur = { 8.33, 25, 66.67 };  
        teikna(tolur);  
    }  
}
```

8 Stoplarit - java

Forritið hér

1	
---	--

Maximum marks: 15

Skrifið aðferðina **finnaUrslit** sem tekur inn í viðfangsbreytu (parameter) fjölda marka sem lið hafa skorað og fjölda marka sem lið hafa fengið á sig, reiknar út stigin og skilar þeim sem fylki sem hefur sama fjölda staka og liðin í keppninni eru mörg.

Eftirfarandi tafla gæti sýnt úrslit leikja. Holland hefur skorað 1 mark gegn Ekvador, 2 mörk gegn Katar og 2 mörk gegn Senegal. Holland hefur fengið á sig 1 mark í leik gegn Ekvador.

	Ekvador	Holland	Katar	Senegal
Ekvador	0	1	2	1
Holland	1	0	2	2
Katar	0	0	0	1
Senegal	2	0	3	0

Þegar reikna á **stigin** er notuð eftirfarandi regla: Ef **jafntefli** er í leik, fær hvort lið eitt stig.

Ef lið skorar fleiri mörk en það fær á sig í leik, fær það lið **3** stig en mótherjinn fær 0 stig

Miðað við töfluna hér að ofan eru stigin þessi:

Ekvador	Holland	Katar	Senegal
4	7	0	6

Afritið eftirfarandi beinagrind í ritilinn

```
public class Bolti {
    public static int[] finnaUrslit(int[][] urslit) {
        int[] nidurstada = new int [urslit.length];
        // forrita hér ***

        // og hingað en ekki lengra ***
        return nidurstada;
    }
    public static void main(String[] args) {
        int[][] markatala = { { 0, 1, 2, 1 }, { 1, 0, 2, 2 },
                               { 0, 0, 0, 1 }, { 2, 0, 3, 0 } };
        int[] urslit = finnaUrslit(markatala);
        System.out.println(Arrays.toString(urslit));
    }
}
```

9 Bolti - Java

Forritið hér

1	
---	--

Maximum marks: 15

Skrifið klasann **Klukkan** sem útfærir óbreytanlega (e. immutable) klukku. Klukkan geymir klukkustund, mínútu og sekúndu. Þegar klukka er **smíðuð (e. constructor)** er athugað hvort klukkustund, mínúta og sekúnda er lögleg. Ef klukkustund, mínúta eða sekúnda eru ekki lögleg fær klukkan gildin 0 kls, 0 mínúta og 0 sekúnda í smiðnum. Þið fáið aðferðina **erLogleg** sem skilar boolean. Þið þurfið ekki að forrita hana.

Klukkan hefur aðferðina **tic** sem smíðar og skilar nýrri klukku sem er einni sekúndu seinna en núverandi klukka. Eftir að klukka er 23 59 59 verður hún 0 0 0. Eftir að klukka er 22 59 59 verður hún 23 0 0

Afritið í ritil

```
public class Klukkan {
    // forritið tilviksbreytur ***

    public Klukkan(int k, int m, int s) {
        // forritið smiðinn ***
    }
    /**
     * Skilar true ef kls (k) er á bilinu 0 til 23 (báðar meðtaldar) og
     * mínúta (m) er á bilinu 0 til 59 (báðar meðtaldar) og sekúnda (s) er
     * á bilinu 0 til 59 (báðar meðtaldar)
     */
    private static boolean erLogleg(int k, int m, int s) {
        // ekki forrita hér - reiknið með að þið fáið útfærsluna
    }

    public Klukkan tic() {
        // forritið hér ***
    }

    public static void main(String[] args) {
        // prófanaaktygi - þið forritið ekki hér
    }
}
```


10 Klukkan - java**Forritið hér**

1	
---	--

Maximum marks: 15

Í þessu dæmi er verið að skoða vinsældir tónlistarkonu yfir tiltekin tímabil. Vinsældirnar eru breytilegar en eru alltaf vaxandi. Vinsældirnar vaxa **línulega** yfir sum tímabil en vaxa með **veldisfalli** yfir önnur. Upplýsingar um þetta eru geymdar í skrá **timabil.txt** og eru skráðar þannig að fyrst kemur **fjöldi** (fjöldi mælinga) sem er heiltala og síðan koma ein eða fleiri línur af þrennd: **vöxtur**, **frá** og **til** þar sem **vöxtur** er strengur (línulegt eða veldis) og **frá** og **til** eru heiltölur:

fjöldi
vöxtur,frá,til

Dæmi um innihald skráar **timabil.txt** gæti verið eftirfarandi:

```
15
línulegt,1,3
veldis,4,10
línulegt,11,15
```

Til upplýsinga er úttakið úr keyrslunni eftirfarandi.

```
[0.0, 0.5, 1.0, 1.5, 17.5, 26.5, 37.5, 50.5, 65.5, 82.5, 101.5, 107.0, 107.5, 108.0, 108.5, 109.0]
```

Skrifið forrit sem les inn gögnin í **timabil.txt** og reiknar út vinsældirnar (**y**) á tímanum 1 til og með **fjöldi**. Línulega fallið er $f(x) = 0.5 \cdot x$ og veldisfallið er $f(x) = x \cdot x$. **vinsaeldir** aðferðin reiknar gildi á $f(x) + y[\text{fra}-1]$ fyrir $x=\text{fra}$ til $x=\text{til}$ og setur í stak $y[x]$

Interface fyrir **Reikna** er eftirfarandi. Ekki þarf að breyta því og ekki afrita í ritil:

```
public interface Reikna {
    double reikna (int x);
}
```

Afritið eftirfarandi beinagrind af **Vinsaeldir** klasanum í ritilinn. Forritið það sem á vantar í **vinsaeldir** og **main** eins og merkt er

```
import java.util.Arrays;
public class Vinsaeldir {
    // Reiknar gildi á  $f(x) + y[\text{fra}-1]$  fyrir  $x=\text{fra}$  til  $x=\text{til}$  og setur í stak  $y[x]$ 
    public static void vinsaeldir(Reikna f, int fra, int til, double[] y) {
        // forritið hér ***

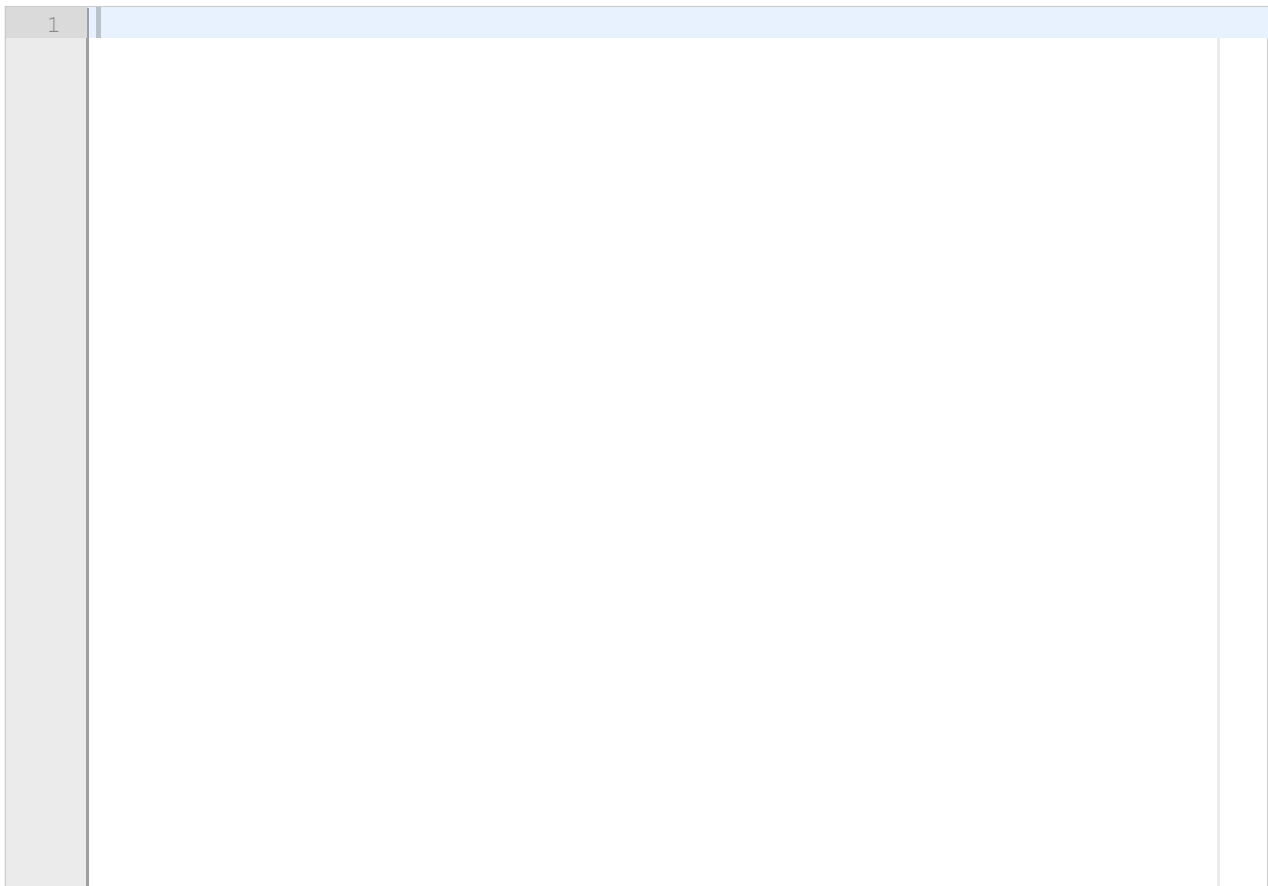
        // og hingað ***
    }

    public static void main(String[] args) {
        In in = new In ("timabil.txt");
        int fjoldiMaelinga = in.readInt();
        double [] y = new double[fjoldiMaelinga+1];
        in.readLine();
        while (in.hasNextLine()) {
            String [] timabil = in.readLine().split(",");
            // forritið hér ***
        }
    }
}
```

```
        // og hingað ***  
    }  
    System.out.println (Arrays.toString(y));  
}  
}
```

11 Vinsældir

Forritið hér



Maximum marks: 15