Oppgåvesamling IT1, 2022-2023

NB: Oppgåvene frå boka (Aschehoug) er òg viktige å kunne løyse. Dei blir ikkje gjentatt i dette dokumentet. Du kan hoppe over oppgåver, men du skal kunne løyse dei. Legg alle besvarelsar på din eigen Github (du får hjelp til dette).

## Oppgåve 1

* Lag eit program som ber brukaren om å skrive inn eit binært tall (eksempelvis 1010). Bruk gjerne prompt.
* Konverter dette til eit heiltall (integer).
* Skriv ut resultatet.

(Hint: parseInt)

## Oppgåve 2

Same som forrige, men no skriv brukaren inn eit hexadesimalt tall (eksempelvis «f»).

## Oppgåve 3

Spør brukaren eit spørsmål. Dersom eit visst ord er inne i setningen så skriv du eit bekreftande svar, som «å, eg og likar …».

(Hint: indexOf eller includes. Bruk gjerne begge.)

## Oppgåve 4

Lag fylgjande oppsummering på nettsida di. NB: Dynamisk. Det vil sei at tala for til dømes bredde, høgde og nettlesarversjon skal endre seg basert på enheten du sit på. Dette kan gjerast på meir eller mindre effektive måtar. Vurder ulike alternativ.  
Et bilde som inneholder tekst

Automatisk generert beskrivelse

Legg til fleire eigenskapar du kan hente ut om du ynskjer. Liste over anbefalte ting å kjenne til:

* Operativsystem (typisk Windows, MacOS, ChromeOS, ulike Linux-variantar el.l.)
* Nettlesar, versjon, «produsent»
* Størrelse på skjerm (høgde, bredde)
* Lokasjon
* … kva meir kan du finne ut?

## Oppgåve 5

Les inn to tal frå brukaren. Summer desse, og skriv ut resultatet.

## Oppgåve 6­

Kva er feil eller dårleg med variabelnavna under? Gje ein kort forklaring og gje dei betre variabelnavn.

1. let Et tall = 123
2. let 3variabelen = 3
3. let Variabel1 = "teksten skal hit"
4. let detførstetallet = "1"
5. let %AvTallet = 15
6. let SummenAvRegnestykketEr = 16

## Oppgåve 7

Skriv eit program som skriver "5 ganger 10 er 50" til konsoll ved å bruke variablar for tala 5, 10 og 50.

## Oppgåve 8

Skriv et program som bruker variablene under (erstatt variabelnavnene med mer passende navn) til å skrive forslag til e-post adresser til konsoll. Be gjerne om input fra brukeren.  
var1 = fornavn  
var2 = etternavn  
var3 = domene

Kriterier:

* Epostadressene skal bestå av fornavn og etternavn, separert med punktum, etterfulgt av @ og et valgfritt domene, for eksempel "test.testesen@bedrift.no".
* Epostadressene som skrives ut skal være uten mellomrom.
* Variabelen som inneholder "domene" skal ikke inneholde "@".
* Epostadressen skal lagres i en egen variabel, og denne nye variabelen skal printes ut

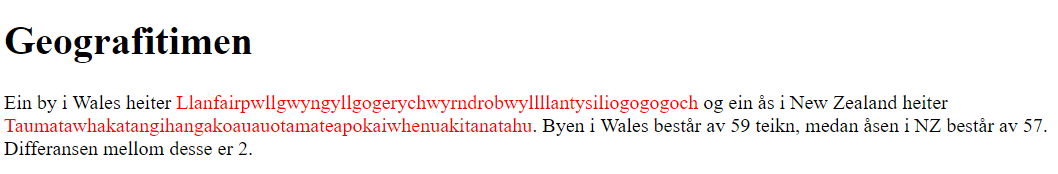
## Oppgåve 9

1. Hvorfor blir ikke regnestykket under regnet ut når console.log-setningen kjøres?  
   console.log("4 + 6");
2. Hvorfor blir to forskjellige tall printet når koden under kjøres?  
   let a = 3.4;  
   let b = 20.5;  
   let total = a \* b;  
   console.log(total);  
   console.log(parseInt(total));
3. Hva er innholdet i variabelen “poeng” etter at denne koden kjøres?  
   let poeng = 0;  
   poeng = poeng + 1;  
   poeng++;  
   poeng += 1;

## Oppgåve 10

1. Hva er feil med koden i "Kodebit 1" under? Rett den opp så den kjører.  
   let gate = 'Kongens gate'  
   let husnr = 432  
   let oppgang = b  
   let adresse = gate + husnr + oppgang  
   console.log(adresse)
2. Skriv adressen ut på følgende to måter ved å bruke variabler: "Adressen er Kongens gate 432b" og "Gaten er Kongens Gate, husnummeret er 432, oppgang b"

## Oppgåve 11

Wales har ein by ved namn Llanfairpwllgwyngyllgogerychwyrndrobwyllllantysiliogogogoch. På New Zealand er det ein ås som har blitt gitt det lange navnet Taumatawhakatangihangakoauauotamateapokaiwhenuakitanatahu. Lag eit program som printer ut antallet bokstaver i kvart namn. Hint: String har ein funksjon som heiter length. Beregn deretter differansen mellom desse to. Få til fylgjande formatering i HTML-dokumentet ditt, og eksperimenter med eigne “vriar” på denne:

## Oppgåve 12: Avlesing av tekstfelt og utrekning

La brukaren skrive inn to eigenvalte (stads)namn og berekn deretter dei same data du gjorde over. Treng du gjere store endringar i programmet ditt? Forsøk gjerne å bruke tekstfelt og ein knapp som i skjermbiletet under. Eventuelt **prompt for å gjere det enklare**. Pass i tillegg på at du unngår minusteikn i utskrifta dersom den første staden har færre antall teikn enn den andre. (Bonusoppgåve: Handter brukarfeil, som til dømes manglande input. Dette tek me opp seinare).  
Et bilde som inneholder tekst

Automatisk generert beskrivelse

# Oppgåve 13: Aldersgrense (if-else)

Lag ei aldersgrense på nettsida di. Spør brukaren kor gamal h\*n er. Dersom over 18 kan du vise eit kattebilete pluss litt tekst, dersom ikkje skal du gje beskjed om at innhaldet dessverre er sperra (el.l.).

Hint: img src=”” er lovleg å bruke. Då vises ikkje eit bilete i heile tatt. Du kan òg skjule og vise HTML-element etter behov, ved å endre på CSS frå JS.

# Oppgåve 14: Biletgalleri (array, if-else m.m.)

Lag eit enkelt biletgalleri vha. array. Meir konkret: Lagre bileta (navnet) i ein array og gjer det mogleg å bytt bilete ved å trykke på ein “neste-knapp”.

Her er det heilt sikkert fristande å finne ein ferdigløysning på nett el.l. Eg anbefalar å lage ein særs enkel versjon basert på det me har lært så langt, for å forstå prinsippet skikkeleg.

Neste steg: Legg til moglegheit til å bla deg tilbake i biletserien.

# Oppgåve 15: Tilfeldige tal

Finn ut korleis du kan generere tilfeldige tal. Hint Math.random og MDN. Her finn du meir generelle og brukandes hjelp enn det som står i boka.

Kva kan me bruke dette til? Kom gjerne med forslag til bruksområder! Lag ditt eige program som benyttar tilfeldige tal.

# Oppgåve 16: Gjettespel

Versjon 1: Grunnleggande

* Lagre eit tal mellom 0-10 i ein variabel. La dette vere tilfeldig.
* Be brukaren om å gjette på eit tal ml 0-10 og lagre dette i ein ny variabel.
* Kontroller om brukaren har gjetta riktig, og gje ein passande tilbakemelding.

Versjon 2: Avansert

* La brukaren få gjette flere gonger dersom h\*n tok feil, og gje hint som «for høgt» eller «for lågt». La brukaren gjette fram til det blir riktig.
* Tell kor mange gonger brukaren har gjetta, eller sett ein begrensing på dette.
* Legg til fleire krav om du vil.

# Oppgåve 17: Førarkort

Lag ein enkel app som kan hjelpe folk med å få vite kva setifikat dei kan ta. Du ber om alderen og presenterer deretter bileter og tekst som fortel kva dei kan ta sertifikat på (meir avansert om du vil: https://lovdata.no/forskrift/2004-01-19-298/§3-1):

1. Yngre enn 16. Ingenting, brukaren er for ung.
2. 16 år: moped
3. 18 år: bil
4. 21 år: lastebil
5. 24 år: buss

Kan du legge inn ei øvre aldersgrense òg? Kanskje som ein beskjed om at du må gjennomføre faste kontrollar dersom du er over eit visst antall år?

# Oppgåve 18: Halloween, skrøymande bra app

Lag ei nettside som har som mål å skrøyme brukaren. Eksempelvis kan du laste inn ei nettside med behagelege fargar, musikk og generelt god stemning. Etter ei tilfeldig tid så kjem det ein høg lyd og eit skrøymande bilete (eller ein video) – samt andre endringar du ser kan vere spanande.

Hint: Timing-funksjonar i JS. setTimeout og/eller setInterval.

# Oppgåve 19: Terningspel

Ta utgangspunkt i 2 terningar, eller fleire. Når sida lastast på nytt, eller når ein trykker på ein knapp, skal terningane få kvar sin verdi mellom 1-6. Skriv ut kva desse er. Deretter kan du til dømes sjekke om alle er like, om dei totalt overstig ein viss sum osv. Gje deg sjølv nokre utfordringar her (også med å vise fram bileter, om du vil).

V1: Enkel versjon:

Et bilde som inneholder tekst

Automatisk generert beskrivelse

V2: Meir avansert versjon:



v3: Bruker løkker og arrays for å generere så mange terningkast som brukaren ynskjer.

# Oppgåve 20: Hjelp kyllingen

Du skal lage deler av ein app for ungar. Nødvendige bileter skal du laste ned sjølv. Appen (nettsida) opnar med at eit bilete av eit egg vises fram, med forklarande tekst til brukaren.

|  |  |
| --- | --- |
| Et bilde som inneholder egg  Automatisk generert beskrivelse  Oppstart. (Manglar informasjon til brukaren, mellom anna.) | Steg/klikk nr. 4: Kylling inne i egg. Manglar elles nokre detaljar frå oppgåvetekst. |

Funksjonalitet vidare, når brukaren trykker på egget:

* Klikk nr. 1: Egget får ei lita sprekke tvers over.
* Klikk nr. 2: Større sprekke.
* Klikk nr. 3: Egget opnar seg endå meir, og du ser deler av ein kylling innanfor egget.
* Klikk nr. 4: Den øvste delen av egget ligg på bakken/golvet, der resten av egget står med kyllingen oppi.
* Klikk nr. 5: Brukaren får beskjed om at ein ikkje kan trykke på egget meir, og at ein er ferdig.

Presenter innhaldet på en oversiktelig og god måte. Bileta skal vere tilpassa på ein fornuftig måte, med tanke på til dømes format, størrelse og oppløysning.

# Oppgåve 21: Gjettespel, extended edition

Et bilde som inneholder tekst

Automatisk generert beskrivelseEt bilde som inneholder tekst

Automatisk generert beskrivelse

## **Grunnfunksjonalitet**

NB: Dette programmet vil etterkvart få meir avanserte krav til funksjonalitet.

Kravspesifikasjon, programmering og design:

* Når spelet startar opp skal ein mynt plasserast bak ein av dei tre posisjonane (grafisk: koppane).
* Deretter er det opp til brukaren å trykke på ein av dei tre koppane, for å gjette kvar han eller ho trur mynten er.
* Programmet ditt skal gje ein tilbakemelding som fylgjer:
* “Gratulerer!” om riktig, og “Dessverre, det var feil!.” om det var feil.
* Dersom ein gjetta feil, men var nært (nabokoppen) skal dette bli lagt til i tilbakemeldinga: “.. men du var nærme!”
* Dersom ein gjetta feil, men var langt vekke (to koppar vekke) skal dette bli lagt til i tilbakemeldinga: “.. og du var langt unna!”
* Tilleggskrav, visuelt:
  + Bruk animasjon for å avsløre kva val som er tatt, og resultatet (til dømes kan du la koppen gradvis flytte seg oppover).
  + Koppen kan bytast ut med mynten dersom ein har gjetta riktig, eventuelt visast delvis under.
  + Det skal vere raud bakgrunnsfarge på tilbakemeldinga (i form av tekst) dersom ein har gjetta feil, og grønt om ein gjettar riktig. Skriftfargen vel du sjølv. Det skal vere lett å lese.
* Etter at ein har gjetta må ein trykke på ein knapp for å starte på nytt. Denne knappen skal ikkje vere synleg før ein har gjennomført ein “gjetting”.
* Det skal ikkje vere mogleg å gjette fleire gonger enn ein, der brukaren skal få tilbakemelding og bli “blokkert” dersom ein forsøker fleire gonger.

# Oppgåve 22: “Chatbot”

Fortset med programmet me byrja på i timane, der forsøker å få det til å sjå ut som om brukaren av programmet har ein samtale med datamaskina. Tips:

* Loggfør all kommunikasjonen i ei punktliste (ul-tag med li-taggar inni). Oppdater denne etterkvart som programmet skrider fram.
* Spør etter navn og sei “Hei, navn!”
* Spør etter interesse/hobby, og følg dette opp i kommunikasjonen. Type “Å, jeg også liker …”. Korleis skal du handtere ulike moglege utfall? Korleis kan du styre kommunikasjonen for å gjere det enklare for deg?
* Viktig: Bruk toUpperCase og toLowerCase for å handtere problematikk med store og små bokstavar, bruk || (or) og && (and) for å kombinere ulike testar, og includes for å leite etter spesifikke ord inne i ein setning (til dømes dersom brukaren svarar med lengre setningar).   
  Avansert: Forsøk å unngå problem med dette at brukaren gjerne kan skrive “jeg liker i hvert fall ikke ski”, og at det blir tolka som at ski er ei interesse.

# Oppgåve 23: Tilfeldige fargar

Lag funksjonar som returnerer ein tilfeldig fargeverdi. Dette kan du til dømes bestemme at skal vere type:

* red, blue, green osv.
* RGB-referansar
* hexadesimale kodar

Slå opp på W3Schools korleis du refererer til fargar i dei ulike “systema”. Kva er lettast å generere tilfeldig synes du?

Bruk den tilfeldige fargen for gjere ein endring på bakgrunnsfargen til body (avansert: ved faste intervall). Skriv til ei kvar tid ut kva som er satt som bakgrunnsfarge. Sjå eksempel her:

Et bilde som inneholder tekst

Automatisk generert beskrivelseEt bilde som inneholder tekst

Automatisk generert beskrivelse

Lag gjerne animasjon av overgangen mellom ein del ulike fargar òg, om du har lyst å gå vidare i pensum.

# Oppgåve 24: OLED-assistent

(Sjå oppgåve 23.)

OLED-skjermar slit med at dei ikkje kan ha eit statisk bilete over lengre tid, då dei kan oppleve «innbrenning» (meir eller mindre permanente pikslar). Du kan til dømes sjå TV2-logoen innbrent oppe i høgre hjørne dersom du ser mykje på den kanalen.

Lag ein app som testar for denne innbrenninga. Funksjonalitet:

* Ved faste intervall skal heile bakgrunnsfargen endre seg.
* Generer tilfeldige fargar og skift mellom desse.
* Få til ein gradvis overgang frå ein farge til ein annan.

Avansert: Lag til eit valg for innstillingar, der brukaren kan bestemme kor ofte fargane skal endre seg.

Finn gjerne på eigne vriar på denne oppgåva.

# Oppgåve 25: Array med tilfeldige tal

Opprett ein array med 100 tilfeldige tal mellom 1-20.

Tell kor mange det er av talet 10.

Tell kor mange som er over 10.

Summer alle tala.

… (fleire testar og krav etter behov.)

# Oppgåve 26: Kryptering

NB: Denne oppgåva kan løysast på fleire ulike måtar, og på meir eller mindre avanserte måtar. Begynn med ein tilnærming som gjer meining for din del. Ikkje forsøk å løys alt på ein gong, få ein og ein enkelt del til å fungere for seg først.

Lag ei nettside som lar deg enkelt leike med å kunne både kryptere og dekryptere meldingar (i tekstform). Me bruker Caesar-kryptering for å oppnå dette, der du kort fortalt ved å kryptere gjer ein forskyvning i alfabetet, der til dømes ein 'a' blir 'b' dersom du har ein forskyvning (nøkkel) satt til 1. Du kan sjølvsagt legge til andre krypteringsmåtar dersom du vil.

Legg inn ei moglegheit for å “brute force” ein kryptert melding, dvs. at du forsøker alle moglege kombinasjonar.

Bruk tid på å planlegge løysinga! Gjerne saman med nokon andre. Går det an å skrive deler av programmet kvar for seg og slå saman?

(Fleire detaljar om krypteringa blir gjennomgått i timane, samt “oppstartshjelp”.)

# Oppgåve 27: Array-repetisjon, “skryteApp”

Det er dårleg stemining i klasserommet. Elevar snakkar stygt til kvarandre. Løysinga blir derfor å lage eit program som gjer som resultat noko som dette:



Forslag til fokus:

* Spør kven ein skal skryte av.
* Spør kor mange skryteord du ynskjer å ta med.
* Skriv ein funksjon som tek inn punkta over og skriv ut resultatet til HTML.
  + Her må/bør du bruke løkke i forbindelse med array.
* Skriv CSS som gjer dette endå meir “koseleg”. Mitt hjarte er animert (det pulserer), klarar du det?
* Ekstra: Sjekk at det ikkje blir bedt om fleire skryteord enn det ligg i arrayen. Sjå òg til at det er minst 1 stk. skryteord.

Samling med eventuelle skryteord (eksempel):

let *skryteOrd* =[

"sjenerøs",

"omgjengelig",

"ambisiøs",

"munter",

"hardtarbeidende",

"troverdig",

"tålmodig",

"optimistisk",

"følsom",

"sosial",

"besluttsom",

"morsom",

"blid",

"vakker",

"lekende",

"elskverdig",

"talentfull",

"begavet"

];

**Alternativ oppgåvetekst:**

Lag ein “troll”-generator. Altså, spy ut spam i ulike diskusjonsforum/kommentarfelt med tilfeldige, provoserande utsagn. NB: Ikkje strekk det for langt, slik at du faktisk gjer dette i praksis.

# Oppgåve 28: Array-repetisjon, “sensureringsApp”

Ta inn ein setning og sensurer bort ord som ligg i ei liste (array) med problematiske ord. Byt ut med noko anna.

Hint: Hugs tidlegare oppgåve på prøve der du luka ut upassande ord.

# Oppgåve 29: Array med objekt: Brukerliste

Opprett ein array med minst 3 ulike objekt, der kvar av desse skal innehalde fylgjande informasjon:

* Navn
* Fødselsår

Lag ein funksjon eller løkke som går gjennom alle elementa og bestemmer alder til kvar person. Skriv berre ut personar under 20 (eller andre liknande krav).

Hint: Bruk «Date»-objektet. let tid = new Date(); Slå opp kva du kan hente ut ved å skrive tid.xxx

# Oppgåve 30: Filmer i array med objekt

1. Lag en nettside for å registrere filmer med følgende informasjon:
   1. Tittel
   2. Regissør
   3. Om du har sett filmen
2. Informasjonen fra nettsiden skal lagres i en array med objekter. Hvert objekt skal beskrive en film med egenskaper for tittel, regissør og om du har sett filmen eller ikke (true/false ).
3. Legg inn 5 filmer som du har sett og 5 filmer som du ikke har sett.
4. Skriv ut alle filmene. Bruk en løkke for å gå gjennom arrayen og skrive filmenes tittel og regissør til nettsiden.
5. Sorter objektene i arrayen etter filmtittel. Skriv ut den sorterte arrayen til nettsiden for å sjekke at de er sortert riktig. Se boken for hjelp til denne.

# Oppgåve 31: Påskerenn

Du skal lage eit program som skal brukast under det årlege påskerennet på stølsområdet Olastøl i Hardanger. Her samlast gode hyttenaboar for å gjennomføre konkurransar i 3 stk. ulike aktivitetar. Prestisje, knekte lårhalsar og forholdsvis godt humør er viktige stikkord.

Lag ein liten introduksjon til påskerennet øverst i dokumentet, gjerne med noko enkel påskegrafikk. Du blir ikkje vurdert på kor bra denne delen ser ut.

**Krav 1**: Deltakerlista varierer frå år til år, men du skal **forhåndsregistrere** 5 stk. deltakarar (hardkoda inn ved oppstart). Typisk bruker du arrays med objekt (eventuelt Map). Informasjonen som MÅ vere med om desse er:

* Navn
* Alder
* Resultat frå aktivitet 1 (endå ikkje nokre data her)
* Resultat frå aktivitet 2 (endå ikkje nokre data her)
* Resultat frå aktivitet 3 (endå ikkje nokre data her)
* Total poengsum, resultatet frå dei tre aktivitetane

Utover dette kan du legge til meir informasjon etter behov. Kva sjølve aktivitetane består av treng du ikkje ta stilling til, om du ikkje vil.

**Krav 2**: Deltakerlista skal skrivast ut til brukaren, ved oppstart, slik at ein ser kven som er med. Typisk ser ein for seg at navn og alder må vere med i denne oversikten. Altså, deler av data frå “krav 1” skal visast ved oppstart.

**Krav 3**: Det skal vere mogleg å registrere nye deltakarar, utover dei forhåndsbestemte. Igjen er navn og alder det som er nødvendig på dette punktet. Etter at ein ny deltaker er registrert skal deltakerlista (frå krav 2) oppdaterast.

**Krav 4**: Det skal vere mogleg å registrere resultat frå konkurransane. Eksempelvis, deltakar nr. 1, “Jo Bjørnar”, har fått 34 poeng under aktivitet nr. 1, 22 p. for akt. 2 og 31 p. for akt. 3. Basert på dette skal data i kolleksjonen oppdaterast, og total poengsum reknast ut.

Du set sjølv begrensingar på maks poengsum (til dømes maksimum 100 poeng per aktivitet) og andre eventuelle reglar/krav.

**Krav 5**: Det skal vere mogleg å vise ein oppsummering av resultata frå konkurransen. Typisk vil ein ved å trykke på ein knapp få ei utskrift som viser resultatlista sortert på den totale poengsummen. Viktig: Det skal berre skrivast ut resultat for deltakarane som har fått ein total poengsum.

Når det gjeld dei tre første plassane skal det visast fram pokalar (eller annan unik grafikk, type bilete/ikon eller liknande) foran desse. Resten av plasseringane bør det stå nummer foran.

[Løysingsforslag](https://github.com/hausnes/it1/tree/main/jobjornar/javascript/l%C3%B8ysningsforslag/p%C3%A5skerenn)

# Oppgåve 32: Munnleg-praktisk eksempel, hopprenn

Du skal lage eit program som reknar ut poengsummar under eit hopprenn. For å gjere det litt enklare ser me for oss at alle operasjonar blir gjort på same skjerm, det vil sei at både inntasting av alle nødvendige verdiar og resultatet kjem same plass.

Hoppreglane er som følger:

* Eit kvart hopp har ein lengde i antall meter, og stilkarakterar frå 5 ulike dommerar.
* Lengdepoenga reknast ut slik at for kvar meter over K-punkt skal ein få poeng i form av meterverdien, pluss ein grunnpoengsum lik 60.
* Formelen blir då lengdePoeng = 60 + meterverdi \* (lengde – K-punkt).  
  Eksempelvis vil du dersom du hoppar 123 meter i ein bakke med K-punkt på 120 meter og meterverdi på 1.8 få 65.4 poeng.
* Stilpoeng reknast ut slik at 5 ulike dommerar gjer sine karakterar mellom 0 og 20, med intervall på 0.5. Det vil sei at ein til dømes kan gje 14.5 og 18, men ikkje 14.23 osv. Når alle har gjort dette skal høgaste og lågaste karakter strykast. Det er altså berre dei 3 karakterane ”i midten” som er med i endeleg poengsum.
* Total poengsum reknast ut ved at du tek lengdepoenga og legg saman med stilpoenga.

# Oppgåve 33: API-kall, createElement og appendChild

**Del 1:**

Bruk <https://randomuser.me/> for å hente ein tilfeldig “person” og opprett deretter ei overskrift med navn, ei lenke/paragraf med e-posten og eit bilete. Bruk createElement og appendChild for å lage HTML-en.

**Del 2:**

Denne gongen ber du brukaren av programmet om å oppgje kva hen er på utkikk etter, til dømes kan du spørre om:

* Kjønn
* Nasjonalitet
* Antall

Deretter utfører du det same API-kallet og opprettar HTML-en basert på dette.

Bruk òg tid på å plassere innhaldet i div-boksar og stilsett dette slik at det ser bra ut vha. CSS.