

Leksjon 5: Strengbehandling, datoer og matriser

Svend Andreas Horgen, IDI Kalvskinn, NTNU

Lærestoffet er utviklet for emnet IINI3003 Webprogrammering med PHP

Resymé: Funksjoner er viktige fordi de gjør det mulig å tenke mer eller mindre overordnet uten å gå ned i detaljer. Hjulet må ikke finnes opp på nytt hver gang. I sær er funksjonene for å behandle strenger, datoer og matriser nyttige. Boka og leksjonen tar for seg konkret bruk. Etter litt teori vil du få introdusert hemmeligheten bak webbaserte julekalendere – noe som er lett å lage med bruk av datofunksjoner og matrisebehandling.

1. HVORFOR FUNKSJONER FØRST NÅ?	2
1.1. HVORFOR IKKE LAGE ALT SELV?	2
1.2. LAG EN FUNKSJON SELV.	3
1.3. ... OG FINN EKVIVALENTEN I PHP-MANUALEN	3
1.4. HVA SKAL DU SØKE ETTER?	5
2. ET STØRRE "CASE" – DIN EGEN JULEKALENDER.....	5
2.1. EGENSKAPER VED VÅR KALENDER.....	6
2.2. STEG 1 – LAG TABELLEN.....	6
2.3. STEG 2 – HVOR LIGGER INFORMASJONEN SOM SKAL VISES?	7
2.4. STEG 3 – VIS BARE RIKTIG INFORMASJON	8
2.5. EKSTRA-FUNKSJONALITET	11

1. Hvorfor funksjoner først nå?

Det er mange måter å lære PHP på. Vi skulle gjerne ha introdusert funksjonsgalleriet til PHP tidligere, men har valgt å la kurset følge boka. Dermed har du nå:

- Fått installert nødvendig programvare og lagd dine egne, enkle PHP-script.
- Fått føle samspillet mellom klient og tjener.
- Lært hvordan variabler kan brukes effektivt i tekststrenger.
- Sett hvordan skjema kan øke interaktiviteten mellom brukeren og nettstedet.
- Lært flere måter for hvordan gjenbruk kan skje i praksis ved å generaliseres kode og innhold.
- Opplevd hvor lett det er å gjøre feil, og lært noen teknikker for hvordan slike kan unngås.

Du har allerede brukt flere innebygde funksjoner, kanskje uten å være klar over det. Datofunksjonen `date()` kan brukes på mange måter. Med behandling av tidspunkt er det mulig å skreddersy løsningene i større grad enn vist så langt. Sammen med strengbehandling, er datofunksjonene et viktig bidrag i PHP-biblioteket. Vil du lage din egen julekalender hvor bare ”lovlige” luker åpner seg? Skal ulik informasjon vises basert på tidspunkt? Boka og leksjonen ser nærmere på ulike bruksområder.

1.1. Hvorfor ikke lage alt selv?

I forrige leksjon så vi hvordan funksjoner kan øke graden av gjenbruk, og lette vedlikeholdet. Vi listet opp noen fordeler med egendefinerte funksjoner. Det er også mange grunner til å bruke innebygde funksjoner, og en bør ikke bruke tid på å lage sin egen variant av en innebygd funksjon uten å ha god grunn for det. Hvorfor bruke innebygde funksjoner?

- Unngå feil – funksjonene er lagd av dyktige utviklere og har ikke (skal ikke ha) bugs.
- God dokumentasjon – den offisielle PHP-manualen forklarer og eksemplifiserer ulike bruksområder, ofte med utgangspunkt i vanlige problemstillinger. Alle webutviklere har hatt behov for å finne hvor lang tid det er til en viss dato. Det er derfor eksempel på nettopp dette og en rekke andre praktiske problemstillinger i manualen.
- Referansene i manualen til andre hyppig brukte funksjoner gjør at en kan lære mer og utvide sitt repertoar jo mer enn slår opp.

Forrige leksjon tok for seg hvordan funksjoner for å finne areal og volum av geometriske figurer kunne lages omtrent slik det passet. Det ble gitt noen råd om hvordan funksjoner bør lages, men du står helt fritt til å gjøre slik du vil. De fleste innebygde funksjoner gjør mye av det samme, noe du fort vil oppdage:

1. De mottar informasjon i form av argumenter.
2. De jobber på eventuell informasjon og utfører noe.
3. De returnerer som regel en verdi (av og til flere gjennom bruk av referanser).

1.2. Lag en funksjon selv...

Oppgave 5-3 i boka er ment å motivere for neste kapittel, og ber leseren om å skrive en rekke funksjoner som opererer på matriser. Hvordan kan en for eksempel (uten å bruke innebygde funksjoner) lage en egendefinert funksjon som finner ut hvor mange elementer en matrise har? Hintet som oppgis er å se på skillet mellom while og foreach. De to løkkene har en vesentlig forskjell – foreach går gjennom alle elementer i en matrise. Dette kan vi bruke til å lage for eksempel denne funksjonen:

```
function antallElementer($matrise){
//egendefinert funksjon som finner antall elementer
//i en matrise uten å bruke innebygde funksjoner.
//Utnytter foreach for å få det til
$antall = 0;
foreach ($matrise as $nøkkel => $verdi)
    $antall ++;
return $antall;
} //slutt antallElementer
```

Med gjennomgangen av matriseelementer telles en variabel opp. Dermed får en returnert riktig antall. Funksjonen kan brukes slik:

```
$byer = array("Bergen", "Oslo", "Trondheim", "Stavanger", "Tromsø");
echo "Det er " . antallElementer($byer) . " elementer i matrisen";
```

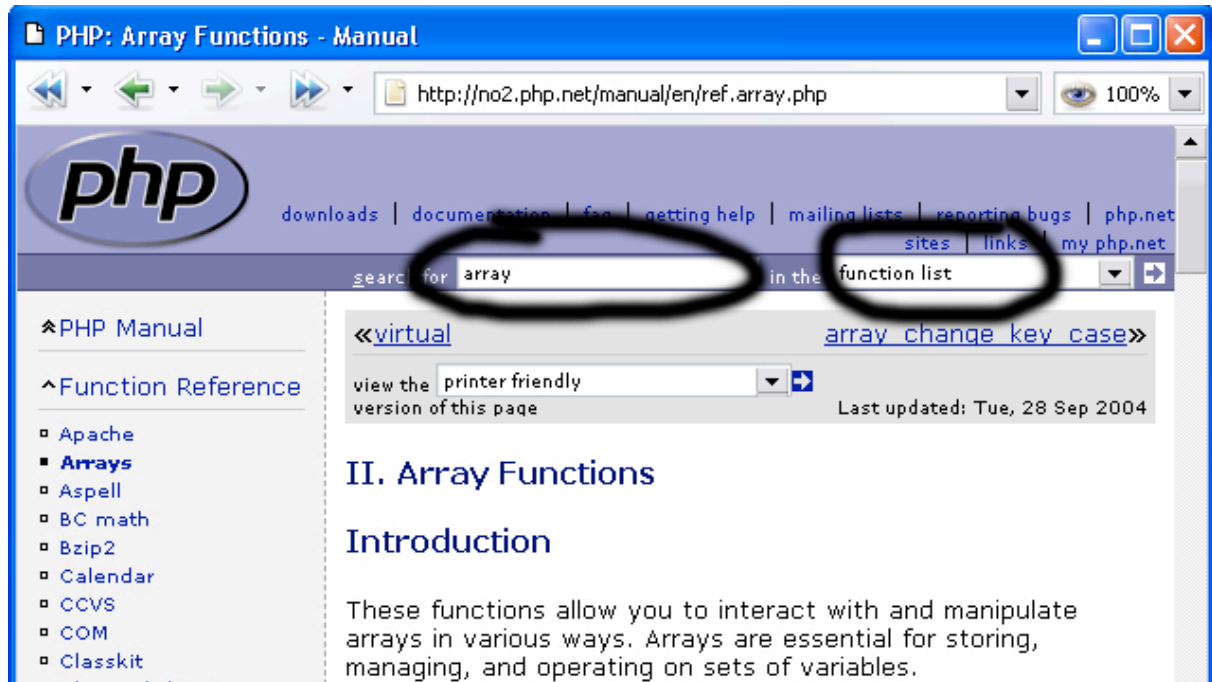
Du kan selv prøve på resten av oppgavene om du ikke allerede har gjort det – det er en fin motivasjon for å sammenlikne med manualen.

1.3. ... og finn ekvivalenten i PHP-manualen

Boka beskriver hvordan manualen kan brukes, og har noen hint om effektiv bruk og hva som er hva. PHP-teamet har fokusert på at det skal være et konsistent grensesnitt over tid. Du vil alltid kjenne deg igjen, og når du lærer deg å slå opp/søke etter informasjon vil du trolig like grensesnittet som ryddig. Dersom du bruker Windows bør du definitivt prøve å laste ned manualen som tradisjonell Windows-hjelp. Da vil du få med noen finesser som gjør søkene enda raskere. Vær da klar over at eventuelle oppdateringer ikke blir med før du laster ned en ny versjon av manualen.

Prøv dette:

- Søk etter *array*, som er engelsk for matrise eller tabell.
- Sørg for at det står ”function list” i nedtrekkslisten bak søkefeltet og teksten ”in the”
- Trykk Enter og du får fram følgende side:



Figur 1: Fra php-manualen på Internett.

- Når et stikkord som tilsvarer en kategori av funksjoner søkes, kommer det fram en introduksjon til hele funksjonsgruppen. Du må bla litt nedover før du finner noe matnyttig. Listen over alle funksjoner har også en kort, beskrivende tekst. Les gjennom fort, og du vil få et kort overblikk over hvordan det er mulig å behandle matriser. Her er et utvalg (nokså tilfeldig valgt):
 - `array_diff` -- Computes the difference of arrays
 - `array_pop` -- Pop the element off the end of array
 - `array_push` -- Push one or more elements onto the end of array
 - `array_rand` -- Pick one or more random entries out of an array
 - `array_reverse` -- Return an array with elements in reverse order
 - `array_splice` -- Remove a portion of the array and replace it with something else
 - `array_sum` -- Calculate the sum of values in an array
 - `array_walk` -- Apply a user function to every member of an array
 - `array` -- Create an array
 - `arsort` -- Sort an array in reverse order and maintain index association
 - `asort` -- Sort an array and maintain index association
 - `count` -- **Count elements in a variable**
 - `in_array` -- Checks if a value exists in an array
 - `sizeof` -- Alias of count()
 - `sort` -- Sort an array

Legg merke til at det om funksjonen `count()` står at den teller elementene i en variabel. Hvis du klikker videre på funksjonsnavnet vil du se at `count()` typisk brukes på matriser. Nest sist i vår oversikt står det at funksjonen `sizeof()` er alias for `count()`. Det betyr at begge gjør det samme – de peker på samme funksjon men har forskjellige navn. Dette letter overgangen fra ulike språk og gjør funksjonsnavnene ekstra lette å huske.

1.4. Hva skal du søke etter?

Av menyen til venstre i bildet over, vil du trolig ikke finne så mange interessante kategorier. Hvis du skal behandle strenger, søker du etter ”strings”, og ”datetime” passer for dato. Ofte har en bare bruk for disse, samt funksjoner for å operere mot databaser og filer. Senere i kurset skal vi se på slike samt grafikkfunksjoner. Gjør deg gjerne kjent med hvilke grupper som fins – slik at du vet det til eventuelle fremtidige behov!

Når du har gjort deg litt kjent med manualen, så les videre i bokas kapittel 6. Legg merke til at funksjonene har beskrivende navn, som ofte vil være på engelsk og gjerne også en programmatisk forkortelse av engelske ord som tilsvarer oppgaven som skal utføres. Et godt eksempel er funksjonen `nl2br()` som mottar en tekststreng og gjør om linjeskift til taggen `
`. Du kan lese funksjonen som ”From new line to (2) break”. Erfaring og dokumentasjonen kan avsløre hva navnene betyr, og gjøre det lettere å huske hva man bør bruke. En fordel med PHP er at mange navn som også brukes i andre språk. Siden C og C++ er utbredte språk, er mange funksjonsnavn hentet derfra. Den noe kryptiske navngivingen er nokså logisk om en bare blir vant å lese mellom bokstavene:

- `strstr()` – står for ”string string”, eller ”Find first occurrence of a string”
- `strlen()` – finne lengden av en streng. ”Get string length”
- `ucfirst()` – første bokstav stor. ”Make a string's first character uppercase”
- `substr()` – Finne en del av en streng, en såkalt sub-streng. ”Return part of a string”.

2. Et større ”case” – din egen julekalender

Boka tar for seg strengbehandling, matriser og datoer i tilstrekkelig detalj i kapittel 6, men legg merke til at det også er smarte triks og tips i kokebok-kapittelet (kap 14). Det repeteres ikke her. La oss heller gi et litt større eksempel som bruker flere av funksjonsgruppene. Du har sikkert sett mange eksempler på mer eller mindre interaktive julekalendere. Ved å lage en slik, kan vi illustrere praktisk bruk av flere ulike funksjoner.

Det første vi må gjøre er å definere oss noen egenskaper som kalenderen skal ha. Den bør ha 24 luker med unikt innhold bak. Hvordan skal kalenderen presenteres? Skal lukene ha tilfeldig plassering (men samme innhold bak hver luke) for to besøk?

Skal det være mulig å åpne alle lukene på en gang, slik noen av oss gjorde når vi var små? Det blir fort kjedelig. Mer salgsorienterte nettsteder ønsker å lokke ”kunden” tilbake hver eneste dag over lang tid, trolig for å øke handelslysten eller presentere ny reklame. For å oppnå jevnlig besøk utover i desember, er det vanlig å sende en autogenerated e-post til ”abonnentene” av julekalenderen, og kanskje lokke med å la en heldig vinner hver dag få en premie, i tillegg til innholdet i dagens luke? For å få det til må en URL med personlig informasjon (innlogging for registrering) oppgis i e-posten. Uavhengig av om besøket er

anonymt eller ikke skal bare riktig luke kunne åpnes. Datofunksjonene blir dermed essensielle for å kunne åpne riktig luke til rett tid, mens `mail()` må brukes for å sende påminnelsen.

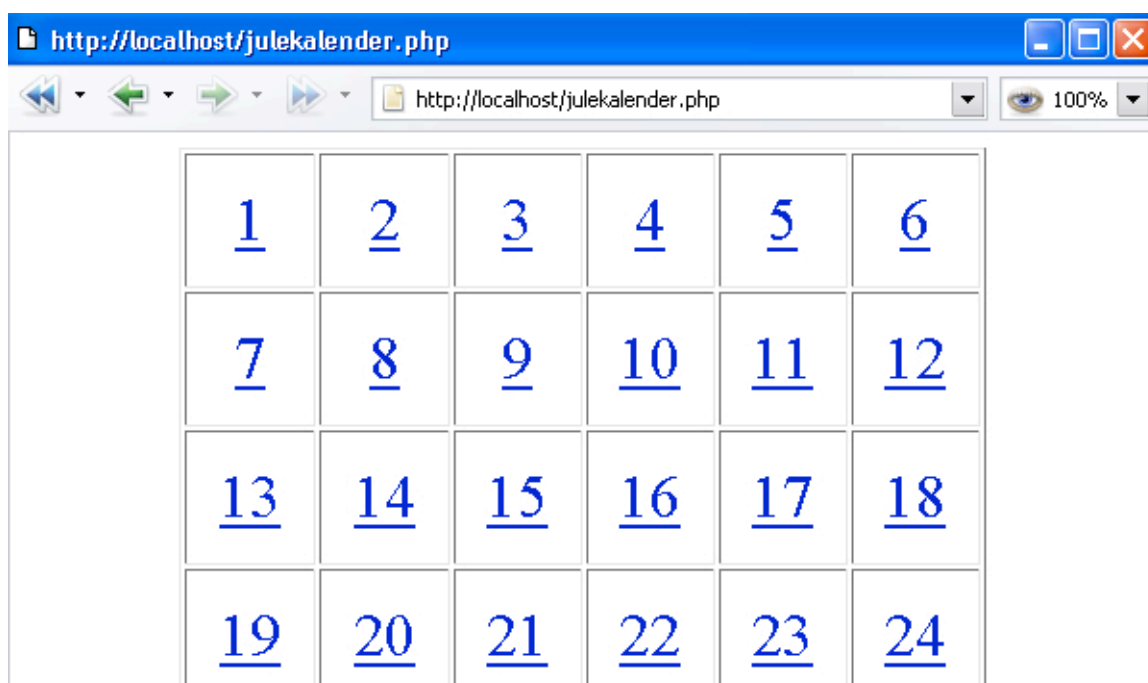
2.1. Egenskaper ved vår kalender

Du har nå sikkert gjort deg noen egne tanker om hvordan en kalender kan se ut. I stedet for å gå inn på prinsipper bak webdesign, definerer vi her kort de egenskaper vi vil at kalenderen skal ha, og ser på prinsippene:

- Selve kalenderen presenteres i en tabell med 4x6 luker. Hele tabellen kan gjerne ha en fin julebakgrunn. Dette tas ikke med her.
- En skal aldri kunne åpne nyere luker enn dagens dato, og ikke jukse på dette ved å stille klokken på sin egen maskin eller lignende.
- Hver celle har et tall – disse er plassert kronologisk utover.
- Når tallet i en luke klikkes, vil innholdet vises. For enkelhetsskyld viser vi bare en ny side, men en flink webdesigner vil lage en mer fancy løsning.
- En bestemt luke har alltid samme innhold.

2.2. Steg 1 – Lag tabellen

Det er enkelt å lage en tabell som den vist her med hardkoding, men mer effektivt med to nøstede løkker. Her er tallene i kronologisk rekkefølge, og du ser likheten med en vanlig julekalender.



<u>1</u>	<u>2</u>	<u>3</u>	<u>4</u>	<u>5</u>	<u>6</u>
<u>7</u>	<u>8</u>	<u>9</u>	<u>10</u>	<u>11</u>	<u>12</u>
<u>13</u>	<u>14</u>	<u>15</u>	<u>16</u>	<u>17</u>	<u>18</u>
<u>19</u>	<u>20</u>	<u>21</u>	<u>22</u>	<u>23</u>	<u>24</u>

Figur 2: En julekalender med 24 luker.

For å få bakgrunn i tabellen, kan `<table background='bildenavn.jpg'>` brukes. Sammen med CSS kan også skrifttypene gjøres mer attraktive, og eventuelt plasseres rundt

omkring uten bruk av tabell. Tabell er derimot enkelt, og når den blir bra nok er det bare å slå av kantene ved å sette `border='0'`.

Løkkene som lager tabellen ser slik ut:

```
<?php /* julekalender.php, versjon 1 */
$tall = 0;
echo "<table border='1' align='center' width='420'>\n";
for ($rad=1;$rad<=4; $rad++){
    echo "<tr>\n";
    for ($kol=1;$kol<=6; $kol++){
        $tall++;
        echo "\t<td height='70'
                width='70'
                valign='center'
                align='middle'>";
        echo "<a href='lenke.html'><font size=6>$tall</font></a>";
        echo "</td>\n";
    }//kolonner
    echo "</tr>\n\n";
}//rader
echo "</table>";
?>
```

Tallene som skrives ut er markert i fet skrift i koden – det er ikke noe hokus pokus med dette. Vi har også satt opp en dummy-lenke for hvert tall.

2.3. Steg 2 – Hvor ligger informasjonen som skal vises?

Neste fokus blir å vise informasjon når en luke klikkes på. Hvor skal denne informasjonen ligge? Her er det flere muligheter:

- I en matrise. Dersom kun ett script brukes kan den ligge øverst i koden. Hvis flere script skal bruke matrisen må den ligge i en fil som inkluderes.
- I 24 filer, der riktig fil kan inkluderes ved behov.

Dersom én av 24 filer velges for inkludering, må en ta hensyn til at brukeren ikke kan gjette filnavnet. En fin mulighet er å la filene som inkluderes ligge utenfor webtreet, dermed er det bare PHP som kan foreta inkludering. En besøkende vil da ikke kunne skrive inn noe filnavn/URL direkte i nettleseren. Vi velger likevel å bruke matrise-notasjonen. Det har en ekstra gevinst når det kommer til det å plassere tallene tilfeldig. Vent og se hvor lett det kan gjøres! Legg først følgende matrise i en include-fil.

```
<?php /* luker.inc.php */
$lukene = array(
    0 => "Her er en feilmelding",
    1 => "Her er teksten til nr 1",
        "Her er teksten til nr 2",
        "Her er teksten til nr 3",
    // alle andre kuttet bort her. De skal normalt være med
        "Her er teksten til nr 23",
        "Her er teksten til nr 24"
```

```
);
?>
```

Du har nå en matrise, og tekstene som skal vises starter med nøkkelen 1. Det vil si at siste element har nøkkel 24, selv om det er 25 elementer i matrisen. Det første elementet har en feilmelding, som vi skal se snart at er fornuftig.

Ved å legge til tre setninger til den opprinnelige koden for å lage tabellen, kan kalenderen bli generert basert på matrisen.

```
<?php /* julekalender.php, versjon 2 */
include "luker.inc.php"; //har matrise med 24 tall og tekster
echo "<table border='1' align='center' width='420'>\n";
for ($rad=1;$rad<=4; $rad++){
    echo "<tr>\n";
    for ($kol=1;$kol<=6; $kol++){
        $neste = key($lukene); //henter neste tall
        next($lukene); //øker den interne pekeren i matrisen
        echo "\t<td height='70'
                                width='70'
                                valign='center'
                                align='middle'>";
        echo "<a href='julekalender_vis.php?luke=$neste'>
                                <font size=6>$neste</font></a>";
        echo "</td>\n";
    } //kolonner
    echo "</tr>\n\n";
} //rader
echo "</table>";
?>
```

Det som er verdt å merke seg er angitt i fet skrift. Funksjonene `key()` og `next()` spiller en viktig rolle her – disse henter ut nøkkelelementene fra matrisen en etter en og øker den interne pekeren etter hvert. Resultatet er akkurat det samme som i bildet vist tidligere, med unntak av at lenken er endret fra en dummy til en som faktisk kan vise informasjonen. Merk at dette kunne vært gjort i samme script, men vi har delt opp i to script for å vise bruken av inkludering (noe som ikke er nødvendig med all kode i ett script). Lenken bak luke 16 blir dermed slik:

julekalender_vis.php?luke=16

2.4. Steg 3 – Vis bare riktig informasjon

Scriptet for å vise informasjon heter altså *julekalender_vis.php* og mottar et lukenummer som argument. Det første som er klart, er at det er matrisen fra include-filen som har informasjonen som skal vises. Riktig element kan hentes ut slik:

```
<?php /* julekalender_vis.php, versjon 1 */
include "luker.inc.php"; //har matrise med 24 tall og tekster+feilmelding
$lukenummer = $_GET['luke'];
echo $lukene[$lukenummer];
?>
```


But behold – thy shall not cheat! Det er ikke lov å jukse. Hvordan kan en besøkende jukse med denne kalenderen? For det første kan alle tall trykkes. Kanskje du tenker at lenkene i setningen

```
echo "<a href='julekalender_vis.php?luke=$neste'>
      <font size=6>$neste</font></a>";
```

ikke burde vært laget med mindre den gjeldende datoen er kommet ”langt nok”? Godt tenkt, men er det nok? Av hensyn til brukervennlighet bør det uansett gjøres slik at bare tall som har en passert dato vises som lenke. Funksjonen `date()` kombinert med en if-setning er nødvendig for å få ut bare riktige tall som understrekede. Legg merke til trikset for å teste (en kan ikke vente på hver enkelt dato for å sjekke at kalenderen faktisk fungerer), og husk å bytte ut kommentaren med burk av datofunksjonen i den endelige løsningen.

```
/* julekalender.php, versjon 3
(det som er bortkuttet skal være samme som versjon2)
*/

//$dagens_dato_som_dag = date("d");
//lurt triks for å teste er å kommentere ut date() og bruke
//et hardkodet tall
$dagens_dato_som_dag = 14;
if ($neste <= $dagens_dato_som_dag ) {
    echo "<a href='julekalender_vis.php?luke=$neste'>
          <font size=6>$neste</font></a>";
}
else {
    echo "<font size=6>$neste</font>";
}
```

Resultatet på den 14. blir at følgende kalender vises:



Figur 3: Kalenderen gir kun mulighet for å åpne dagens og foregående dagers luker. Her er vi kommet til 14. desember.

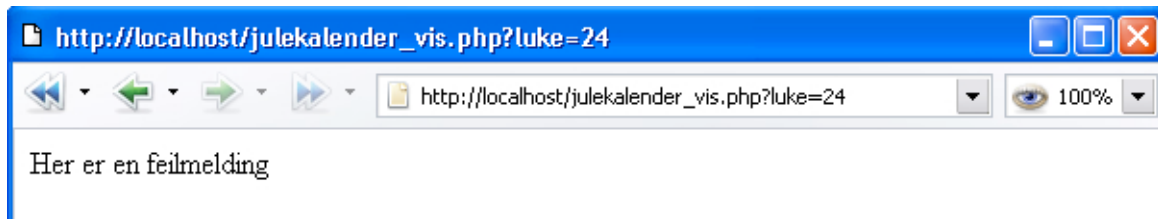
Julenissen sier fortsatt: "But behold – thy shall not cheat!" Hvorfor det? Jo, for slik løsningen er nå med scriptet med siste modifikasjon og *julekalender_vis.php*, er det bare for den besøkende å skrive inn

`julekalender_vis.php?luke=24`

i adressestrengen for å få se siste lukes innhold. Det er nødvendig å foreta en liknende datosjekk også i *julekalender_vis.php*.

```
<?php /* julekalender_vis.php, versjon 2 */
include "luker.inc.php"; //har matrise med 24 tall og tekster
//$dagens_dato_som_dag = date("d");
$dagens_dato_som_dag = 14;
if ( $_GET['luke'] <= $dagens_dato_som_dag ) {
    $lukenummer = $_GET['luke'];
}
else {
    $lukenummer = 0; //feilmelding
}
echo $lukene[$lukenummer];
?>
```

Resultatet blir at når brukeren manipulerer spørrestrengen på for eksempel den 14., fås feilmeldingen opp. Merk at denne løsningen ikke tar hensyn til andre måneder enn desember. Du bør selvsagt legge inn tester for å kun åpne kalenderen mot juletider!



Figur 4: Feilmelding dersom noen prøver å lure til seg en dato frem i tid.

2.5. Ekstra-funksjonalitet

Vi har nå laget en fullgod kalender, og dersom du går gjennom eksempelet raskt en gang til vil du se at det er relativt kurant. Alt bygger på bruk av matriser og datoer. Funksjonene for å behandle disse er godt dokumentert i manualen, og en bør tenke som så at det fins sikkert en funksjon som kan gjøre oppgaven som trengs. Et faremoment er å se seg blind på hvordan et problem skal løses – og så gjøre alt veldig tungvint i stedet for å bruke funksjoner som gjør mye i en operasjon. Utfordringen er ofte å tenke litt alternativt, og våge å prøve nye tanker. Eksperimenter, få erfaring med bruk av de ulike funksjonene, og du vil oppleve at kodingen går mye lettere!

Nå er tallene i kalenderen plassert i rekkefølge, men det er ikke noe i veien for å plassere disse hultert til bultert. Funksjoner som `shuffle()` kan brukes for å stokke om på tallene i en matrise. Du kan derimot ikke bruke `shuffle()` direkte på luke-matrisen, fordi det er bare verdiene som stokkes om (uten at nøklene tas med). Løsningen er å bruke en hjelpematrise `$hjelpematrise` som i neste kodesnutt og så ta `shuffle()` på den. Da får du en matrise med tall som er hultert til bultert. De egentlige lukeverdiene ligger jo i en annen matrise, og skal bare vises når riktig lenke klikkes på. Det eneste vi må gjøre er derfor å sørge for at tallene skrives ut i tabellen i hultert til bultert, og nøyaktig en gang.

Løsningen er usannsynlig enkel. Gå gjennom hjelpematrisen fra start til slutt (den er jo stokket om og dermed er tallene også i tilfeldig rekkefølge). Dette vil være nok!

```
<?php /* julekalender.php, versjon shuffle */
$hjelpematrise = range(1, 24); //lager hjelpematrise
shuffle($hjelpematrise);

echo "<table border='1' align='center' width='420'>\n";
for ($rad=1;$rad<=4; $rad++){
    echo "<tr>\n";
    for ($kol=1;$kol<=6; $kol++){
        //henter neste tall og øker den interne pekeren
        $neste = next($hjelpematrise);
        echo "\t<td height='70'
                width='70'
                valign='center'
                align='middle'>";
        echo "<a href='julekalender_vis.php?luke=$neste'>
                <font size=6>$neste</font></a>";
        echo "</td>\n";
    } //kolonner
```

```

        echo "</tr>\n\n";
    }//rader
    echo "</table>";
?>

```



<u>19</u>	<u>23</u>	<u>8</u>	<u>15</u>	<u>3</u>	<u>16</u>
<u>21</u>	<u>10</u>	<u>4</u>	<u>7</u>	<u>9</u>	<u>1</u>
<u>2</u>	<u>17</u>	<u>24</u>	<u>13</u>	<u>12</u>	<u>20</u>
<u>6</u>	<u>5</u>	<u>11</u>	<u>14</u>	<u>18</u>	

Figur 5: Rekkefølgen endres hver gang siden besøkes. Merk at tallene kun forekommer en gang.

Som du ser er det lett å lage en matrise med tallene 1-24 ved hjelp av `range()` funksjonen. Denne matrisen stokkes om, og da er det bare å hente ut et og et element fra matrisen. Vi har to for-løkker nøstet i hverandre. Den ene går 4 ganger, og den andre 6, totalt $4 * 6 = 24$ ganger. Dette tilsvarer antall elementer i matrisen (1-24 = 24 elementer). Funksjonen `next()` henter ut neste verdi. Vi kan altså være sikker på at hvert element (dvs tall) kommer i tilfeldig rekkefølge for hver gang siden gjenoppfriskes, og samtidig være sikker på at hvert element bare forekommer én gang.

I prinsippet kan det også sendes ut en autogenerated e-post hver dag med påminnelse til de som vil det om at nå er det på tide å åpne neste luke. Normalt vil en liste av e-postadresser ligge lagret på fil eller i en database. Du vil vite nok til å kunne lage løsningen basert på fil/database-innhold om noen få leksjoner. Det er lett å sende en e-post ved hjelp av funksjonen `mail()` men problemet er at noen eller noe må aktivere scriptet som sender ut e-posten hver dag, og du må forhindre at det sendes ut mer enn en gang. Det beste er om du selv har e-post-scriptet på et sted som ingen andre vil nå tak i, og starter det hver dag. Det er også mulig å automatisere dette. Har du for eksempel webtjeneren din på Linux, kan du lage en kronjobb som kjører kl 09:00 hver dag i desember og sender ut til alle påmeldte.