CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 30/08/2024 - 11:20:57.0
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Operações com datas VIVXh89a V: 1.02
76001 CARLOS EDUARDO ANTAO
24FRO404 - 1 _____ / _____ / _____

Operações com datas

Mais um exercício para desenvolver as habilidades até agora trabalhadas, a saber: HTML, CSS e finalmente a programação Javascript. O aplicativo é para fazer cálculos envolvendo datas.

Mas para isso, vamos trabalhar alguns conceitos em JS, vejamos

Função de inicialização Se você precisar executar algo uma única vez ao carregar a página, um caminho possível é

- Definir `<body onload="inicializa()">` que informa que quando a página for carregada, deve ser executada a função `inicializa`.
- No script correspondente, defina a função `inicializa` fazendo o que deve ser feito.

Separar elementos de uma lista Supondo uma string com elementos separados por barras o comando `xx=lista.split('/')` vai criar uma lista onde cada elemento da string original será um elemento da lista

Converter lista de strings em lista de números Agora, o passo seguinte é converter cada elemento em um número. O comando é `yy=xx.map(Number)`. A função `map` aplica a função citada (neste caso é a `Number`) a cada elemento da lista.

Lista de saída No exercício desta folha você eventualmente precisa definir uma data de saída. A solução é

```
resp=' '
resp=dia+'/'+mes+"/"+ano
```

Note que o JS é suficientemente moderno para entender que valor é numérico mas como há strings na expressão (`"/"`) o sinal de adição não é adição e sim concatenação.

A encomenda

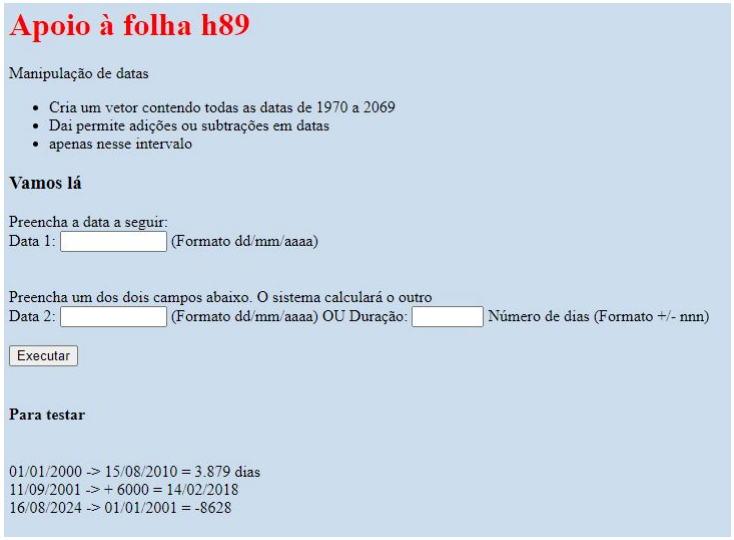
Note que a primeira coisa que o aplicativo pede é a digitação de uma data, em formato válido (`dd/mm/aaaa`) e compreendida no período válido (entre 1971 e 2069).

A seguir, o usuário pode escolher uma de duas opções

- Se optar por uma segunda data, o aplicativo vai calcular quantos dias separam as duas datas
- Se optar por uma quantidade (positiva ou negativa) de dias o sistema vai calcular a data obtida somando ou subtraindo a quantidade de dias informados em relação à primeira data informada.

A estratégia Devido à irregularidade do calendário, é muito difícil (e portanto sujeito a erros) a manipulação direta das variáveis dia, mês e ano em processos de adição e subtração de datas.

Analise a tela do seu aplicativo



Uma boa estratégia (certamente não a única) é criar um vetor de dias, onde o valor anterior corresponde à data de ontem, o atual é o dia de hoje e o próximo é amanha. Vamos chamar este vetor de vetor cronológico. Tendo construído este vetor (que é fixo e não precisa ser recalculado a cada momento), somar e subtrair datas se transforma no problema muito mais simples de somar e subtrair índices dentro desse vetor.

Guardar uma data em um inteiro Posto que uma data é um conjunto de 3 valores (dia, mês e ano, os 3 inteiros), para simplificar a construção do vetor cronológico, é conveniente combinar uma maneira de guardar os 3 valores em um único inteiro. Isto pode ser feito aplicando a seguinte expressão:

$$data = (dia \times 1000000) + (mes \times 10000) + ano$$

Para a construção do vetor cronológico é preciso estabelecer (combinar) com todos os que o usarão, qual o dia inicial do vetor (ou seja qual o valor de vetor[0]). Nada poderá ser feito com datas anteriores a esta. Nesta folha, vai-se usar como data inicial 1^o de janeiro de 1970, que vem a ser a data inicial de várias versões do sistema operacional Unix. Esta folha também vai realizar operações em 100 anos a partir dessa data, indo portanto até o ano de 2070. Diante disso, o tamanho do vetor cronológico é de mais ou menos 36600 elementos ($366 \times 100 = 36600$).

Próximo dia A função chave para construir o vetor cronológico é escrever uma função de nome amanhã, que receba uma data válida e devolva a data que corresponde a amanhã.

```
function amanha(hoje){
 dia=Math.floor(hoje/1000000)
 mes=Math.floor((hoje-(dia*1000000))/10000)
 ano=Math.floor(hoje-((dia*1000000)+(mes*10000)))
 if ((dia==31)&&(mes==12)){return 1010000+ano+1;}
 ud=0
 if ((dia==31)&&((mes==1)||(mes==3)||(mes==5)||
      (mes==7)||(mes==8)||(mes==10))){ud=1}
 if ((dia==30)&&((mes==4)||(mes==6)||(mes==9)||
      (mes==11))){ud=1}
 if ((ano%4==0)&&((ano%100!=0)||(ano%400==0))){
     bis=1}
 else {
     bis=0}
 if ((dia==28)&&(mes==2)&&(bis==0)){ud=1}
 if ((dia==29)&&(mes==2)&&(bis==1)){ud=1}
 if (ud==1){
    return 1000000+(10000*(mes+1))+ano
 }
 else {
    return (1000000*(dia+1))+(10000*mes)+ano
 }
}
```

Agora a criação e preenchimento do vetor cronológico

```
let vd = new Array(36600).fill(0)
function montavet(){
   vd[0]=01011970
   i=0
   while (i<36600){
      vd[i+1]=amanha(vd[i])
   i++
   }
}
```

A manipulação das datas no aplicativo é

```
function execu(){
   dt1=document.getElementById('dt1').value;
   dt2=document.getElementById('dt2').value;
   dur=document.getElementById('dur').value;
   d1=dt1.split('/')
   d1=d1.map(Number)
   x1=(d1[0]*1000000)+(d1[1]*10000)+d1[2]
   i1=vd.indexOf(x1)
   if (dur==""){
      d2=dt2.split('/')
      d2=d2.map(Number)
      x2=(d2[0]*1000000)+(d2[1]*10000)+d2[2]
   i2=vd.indexOf(x2)
   console.log('i2='+i2)
   du=i2-i1
   document.getElementById('dur').value=du
      }
 else {
      i2=i1+parseInt(dur)
      df=Math.floor(vd[i2]/1000000)
      mf=Math.floor((vd[i2]-(df*1000000))/10000)
      af=vd[i2]%10000
   document.getElementById('dt2').value=
           df+'/'+mf+'/'+af
   }
}
```

☞ Para você fazer

Escreva o aplicativo aqui descrito. Mostre para o professor. Atente que haverá solicitações *ad hoc* durante a avaliação. Para testar seu aplicativo, use os exemplos dados na folha, ou quaisquer outros.

Avaliação:

Depois que programar o aplicativo calcule quantos dias há entre 17/12/1980 e 22/ 7/1984 .

Dada a data 14/ 2/1995 qual a data obtida depois de 1977 dias?

Responda aqui:

Dias de diferença	Data achada

404-76001 - /

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 30/08/2024 - 11:20:57.0
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Operações com datas VIVXh89a V: 1.02
76199 DANILO DEPETRIS SOARES
24FRO404 - 2 _____ / _____ / _____

Operações com datas

Mais um exercício para desenvolver as habilidades até agora trabalhadas, a saber: HTML, CSS e finalmente a programação Javascript. O aplicativo é para fazer cálculos envolvendo datas.

Mas para isso, vamos trabalhar alguns conceitos em JS, vejamos

Função de inicialização Se você precisar executar algo uma única vez ao carregar a página, um caminho possível é

- Definir `<body onload="inicializa()">` que informa que quando a página for carregada, deve ser executada a função `inicializa`.
- No script correspondente, defina a função `inicializa` fazendo o que deve ser feito.

Separar elementos de uma lista Supondo uma string com elementos separados por barras o comando `xx=lista.split('/')` vai criar uma lista onde cada elemento da string original será um elemento da lista

Converter lista de strings em lista de números Agora, o passo seguinte é converter cada elemento em um número. O comando é `yy=xx.map(Number)`. A função `map` aplica a função citada (neste caso é a `Number`) a cada elemento da lista.

Lista de saída No exercício desta folha você eventualmente precisa definir uma data de saída. A solução é

```
resp=' '
resp=dia+'/'+mes+"/"+ano
```

Note que o JS é suficientemente moderno para entender que valor é numérico mas como há strings na expressão (`"/"`) o sinal de adição não é adição e sim concatenação.

A encomenda

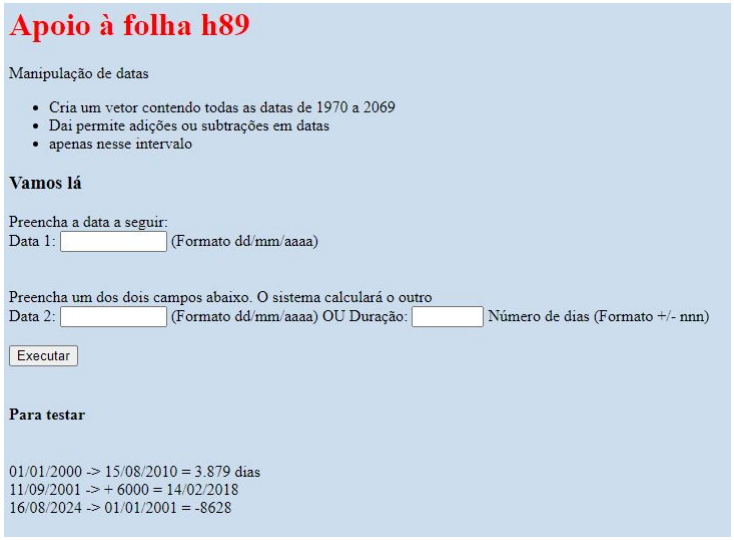
Note que a primeira coisa que o aplicativo pede é a digitação de uma data, em formato válido (`dd/mm/aaaa`) e compreendida no período válido (entre 1971 e 2069).

A seguir, o usuário pode escolher uma de duas opções

- Se optar por uma segunda data, o aplicativo vai calcular quantos dias separam as duas datas
- Se optar por uma quantidade (positiva ou negativa) de dias o sistema vai calcular a data obtida somando ou subtraindo a quantidade de dias informados em relação à primeira data informada.

A estratégia Devido à irregularidade do calendário, é muito difícil (e portanto sujeito a erros) a manipulação direta das variáveis dia, mês e ano em processos de adição e subtração de datas.

Analise a tela do seu aplicativo



Uma boa estratégia (certamente não a única) é criar um vetor de dias, onde o valor anterior corresponde à data de ontem, o atual é o dia de hoje e o próximo é amanha. Vamos chamar este vetor de vetor cronológico. Tendo construído este vetor (que é fixo e não precisa ser recalculado a cada momento), somar e subtrair datas se transforma no problema muito mais simples de somar e subtrair índices dentro desse vetor.

Guardar uma data em um inteiro Posto que uma data é um conjunto de 3 valores (dia, mês e ano, os 3 inteiros), para simplificar a construção do vetor cronológico, é conveniente combinar uma maneira de guardar os 3 valores em um único inteiro. Isto pode ser feito aplicando a seguinte expressão:

$$data = (dia \times 1000000) + (mes \times 10000) + ano$$

Para a construção do vetor cronológico é preciso estabelecer (combinar) com todos os que o usarão, qual o dia inicial do vetor (ou seja qual o valor de vetor[0]). Nada poderá ser feito com datas anteriores a esta. Nesta folha, vai-se usar como data inicial 1^o de janeiro de 1970, que vem a ser a data inicial de várias versões do sistema operacional Unix. Esta folha também vai realizar operações em 100 anos a partir dessa data, indo portanto até o ano de 2070. Diante disso, o tamanho do vetor cronológico é de mais ou menos 36600 elementos ($366 \times 100 = 36600$).

Próximo dia A função chave para construir o vetor cronológico é escrever uma função de nome amanhã, que receba uma data válida e devolva a data que corresponde a amanhã.

```
function amanha(hoje){
 dia=Math.floor(hoje/1000000)
 mes=Math.floor((hoje-(dia*1000000))/10000)
 ano=Math.floor(hoje-((dia*1000000)+(mes*10000)))
 if ((dia==31)&&(mes==12)){return 1010000+ano+1;}
 ud=0
 if ((dia==31)&&((mes==1)||(mes==3)||(mes==5)||
      (mes==7)||(mes==8)||(mes==10))){ud=1}
 if ((dia==30)&&((mes==4)||(mes==6)||(mes==9)||
      (mes==11))){ud=1}
 if ((ano%4==0)&&((ano%100!=0)||(ano%400==0))){
     bis=1}
 else {
     bis=0}
 if ((dia==28)&&(mes==2)&&(bis==0)){ud=1}
 if ((dia==29)&&(mes==2)&&(bis==1)){ud=1}
 if (ud==1){
    return 1000000+(10000*(mes+1))+ano
 }
 else {
    return (1000000*(dia+1))+(10000*mes)+ano
 }
}
```

Agora a criação e preenchimento do vetor cronológico

```
let vd = new Array(36600).fill(0)
function montavet(){
   vd[0]=01011970
   i=0
   while (i<36600){
      vd[i+1]=amanha(vd[i])
   i++
   }
}
```

A manipulação das datas no aplicativo é

```
function execu(){
   dt1=document.getElementById('dt1').value;
   dt2=document.getElementById('dt2').value;
   dur=document.getElementById('dur').value;
   d1=dt1.split('/')
   d1=d1.map(Number)
   x1=(d1[0]*1000000)+(d1[1]*10000)+d1[2]
   i1=vd.indexOf(x1)
   if (dur==""){
      d2=dt2.split('/')
      d2=d2.map(Number)
      x2=(d2[0]*1000000)+(d2[1]*10000)+d2[2]
   i2=vd.indexOf(x2)
   console.log('i2='+i2)
   du=i2-i1
   document.getElementById('dur').value=du
      }
 else {
      i2=i1+parseInt(dur)
      df=Math.floor(vd[i2]/1000000)
      mf=Math.floor((vd[i2]-(df*1000000))/10000)
      af=vd[i2]%10000
   document.getElementById('dt2').value=
           df+'/'+mf+'/'+af
   }
}
```

☞ Para você fazer

Escreva o aplicativo aqui descrito. Mostre para o professor. Atente que haverá solicitações *ad hoc* durante a avaliação. Para testar seu aplicativo, use os exemplos dados na folha, ou quaisquer outros.

Avaliação:

Depois que programar o aplicativo calcule quantos dias há entre 13/ 6/1990 e 6/ 5/1995 .

Dada a data 17/ 3/1979 qual a data obtida depois de 1226 dias?

Responda aqui:

Dias de diferença	Data achada

404-76199 - /

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 30/08/2024 - 11:20:57.0
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Operações com datas VIVXh89a V: 1.02
76018 EDUARDO CAVALHEIRO DIAS
24FRO404 - 3 _____ / _____ / _____

Operações com datas

Mais um exercício para desenvolver as habilidades até agora trabalhadas, a saber: HTML, CSS e finalmente a programação Javascript. O aplicativo é para fazer cálculos envolvendo datas.

Mas para isso, vamos trabalhar alguns conceitos em JS, vejamos

Função de inicialização Se você precisar executar algo uma única vez ao carregar a página, um caminho possível é

- Definir `<body onload="inicializa()">` que informa que quando a página for carregada, deve ser executada a função `inicializa`.
- No script correspondente, defina a função `inicializa` fazendo o que deve ser feito.

Separar elementos de uma lista Supondo uma string com elementos separados por barras o comando `xx=lista.split('/')` vai criar uma lista onde cada elemento da string original será um elemento da lista

Converter lista de strings em lista de números Agora, o passo seguinte é converter cada elemento em um número. O comando é `yy=xx.map(Number)`. A função `map` aplica a função citada (neste caso é a `Number`) a cada elemento da lista.

Lista de saída No exercício desta folha você eventualmente precisa definir uma data de saída. A solução é

```
resp=' '
resp=dia+'/'+mes+"/"+ano
```

Note que o JS é suficientemente moderno para entender que valor é numérico mas como há strings na expressão (`"/"`) o sinal de adição não é adição e sim concatenação.

A encomenda

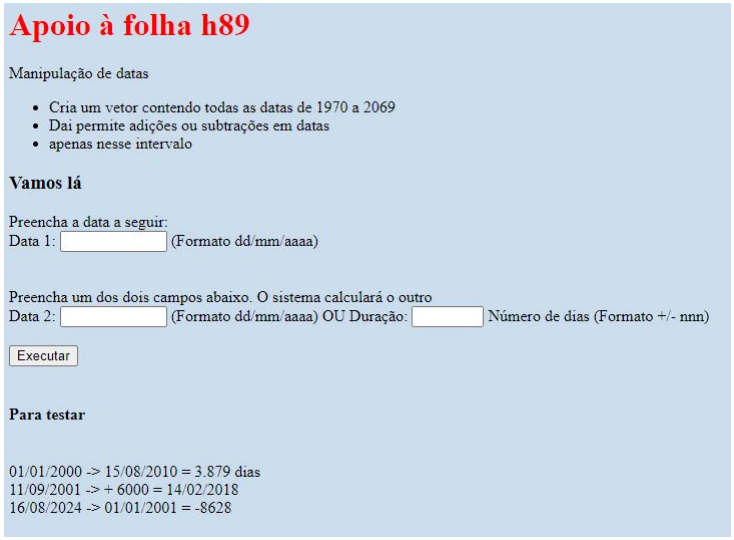
Note que a primeira coisa que o aplicativo pede é a digitação de uma data, em formato válido (`dd/mm/aaaa`) e compreendida no período válido (entre 1971 e 2069).

A seguir, o usuário pode escolher uma de duas opções

- Se optar por uma segunda data, o aplicativo vai calcular quantos dias separam as duas datas
- Se optar por uma quantidade (positiva ou negativa) de dias o sistema vai calcular a data obtida somando ou subtraindo a quantidade de dias informados em relação à primeira data informada.

A estratégia Devido à irregularidade do calendário, é muito difícil (e portanto sujeito a erros) a manipulação direta das variáveis dia, mês e ano em processos de adição e subtração de datas.

Analise a tela do seu aplicativo



Uma boa estratégia (certamente não a única) é criar um vetor de dias, onde o valor anterior corresponde à data de ontem, o atual é o dia de hoje e o próximo é amanha. Vamos chamar este vetor de vetor cronológico. Tendo construído este vetor (que é fixo e não precisa ser recalculado a cada momento), somar e subtrair datas se transforma no problema muito mais simples de somar e subtrair índices dentro desse vetor.

Guardar uma data em um inteiro Posto que uma data é um conjunto de 3 valores (dia, mês e ano, os 3 inteiros), para simplificar a construção do vetor cronológico, é conveniente combinar uma maneira de guardar os 3 valores em um único inteiro. Isto pode ser feito aplicando a seguinte expressão:

$$data = (dia \times 1000000) + (mes \times 10000) + ano$$

Para a construção do vetor cronológico é preciso estabelecer (combinar) com todos os que o usarão, qual o dia inicial do vetor (ou seja qual o valor de vetor[0]). Nada poderá ser feito com datas anteriores a esta. Nesta folha, vai-se usar como data inicial 1^o de janeiro de 1970, que vem a ser a data inicial de várias versões do sistema operacional Unix. Esta folha também vai realizar operações em 100 anos a partir dessa data, indo portanto até o ano de 2070. Diante disso, o tamanho do vetor cronológico é de mais ou menos 36600 elementos ($366 \times 100 = 36600$).

Próximo dia A função chave para construir o vetor cronológico é escrever uma função de nome amanhã, que receba uma data válida e devolva a data que corresponde a amanhã.

```
function amanha(hoje){
 dia=Math.floor(hoje/1000000)
 mes=Math.floor((hoje-(dia*1000000))/10000)
 ano=Math.floor(hoje-((dia*1000000)+(mes*10000)))
 if ((dia==31)&&(mes==12)){return 1010000+ano+1;}
 ud=0
 if ((dia==31)&&((mes==1)||(mes==3)||(mes==5)||
      (mes==7)||(mes==8)||(mes==10))){ud=1}
 if ((dia==30)&&((mes==4)||(mes==6)||(mes==9)||
      (mes==11))){ud=1}
 if ((ano%4==0)&&((ano%100!=0)||(ano%400==0))){
     bis=1}
 else {
     bis=0}
 if ((dia==28)&&(mes==2)&&(bis==0)){ud=1}
 if ((dia==29)&&(mes==2)&&(bis==1)){ud=1}
 if (ud==1){
    return 1000000+(10000*(mes+1))+ano
 }
 else {
    return (1000000*(dia+1))+(10000*mes)+ano
 }
}
```

Agora a criação e preenchimento do vetor cronológico

```
let vd = new Array(36600).fill(0)
function montavet(){
   vd[0]=01011970
   i=0
   while (i<36600){
      vd[i+1]=amanha(vd[i])
   i++
   }
}
```

A manipulação das datas no aplicativo é

```
function execu(){
   dt1=document.getElementById('dt1').value;
   dt2=document.getElementById('dt2').value;
   dur=document.getElementById('dur').value;
   d1=dt1.split('/')
   d1=d1.map(Number)
   x1=(d1[0]*1000000)+(d1[1]*10000)+d1[2]
   i1=vd.indexOf(x1)
   if (dur==""){
      d2=dt2.split('/')
      d2=d2.map(Number)
      x2=(d2[0]*1000000)+(d2[1]*10000)+d2[2]
   i2=vd.indexOf(x2)
   console.log('i2='+i2)
   du=i2-i1
   document.getElementById('dur').value=du
      }
 else {
      i2=i1+parseInt(dur)
      df=Math.floor(vd[i2]/1000000)
      mf=Math.floor((vd[i2]-(df*1000000))/10000)
      af=vd[i2]%10000
   document.getElementById('dt2').value=
           df+'/'+mf+'/'+af
   }
}
```

☞ Para você fazer

Escreva o aplicativo aqui descrito. Mostre para o professor. Atente que haverá solicitações *ad hoc* durante a avaliação. Para testar seu aplicativo, use os exemplos dados na folha, ou quaisquer outros.

Avaliação:

Depois que programar o aplicativo calcule quantos dias há entre 23/ 7/1977 e 22/ 2/1982 .

Dada a data 5/ 7/2000 qual a data obtida depois de 1979 dias?

Responda aqui:

Dias de diferença	Data achada

404-76018 - /

Operações com datas

Mais um exercício para desenvolver as habilidades até agora trabalhadas, a saber: HTML, CSS e finalmente a programação Javascript. O aplicativo é para fazer cálculos envolvendo datas.

Mas para isso, vamos trabalhar alguns conceitos em JS, vejamos

Função de inicialização Se você precisar executar algo uma única vez ao carregar a página, um caminho possível é

- Definir `<body onload="inicializa()">` que informa que quando a página for carregada, deve ser executada a função `inicializa`.
- No script correspondente, defina a função `inicializa` fazendo o que deve ser feito.

Separar elementos de uma lista Supondo uma string com elementos separados por barras o comando `xx=lista.split('/')` vai criar uma lista onde cada elemento da string original será um elemento da lista

Converter lista de strings em lista de números Agora, o passo seguinte é converter cada elemento em um número. O comando é `yy=xx.map(Number)`. A função `map` aplica a função citada (neste caso é a `Number`) a cada elemento da lista.

Lista de saída No exercício desta folha você eventualmente precisa definir uma data de saída. A solução é

```
resp=' '
resp=dia+'/'+mes+"/"+ano
```

Note que o JS é suficientemente moderno para entender que valor é numérico mas como há strings na expressão (`"/"`) o sinal de adição não é adição e sim concatenação.

A encomenda

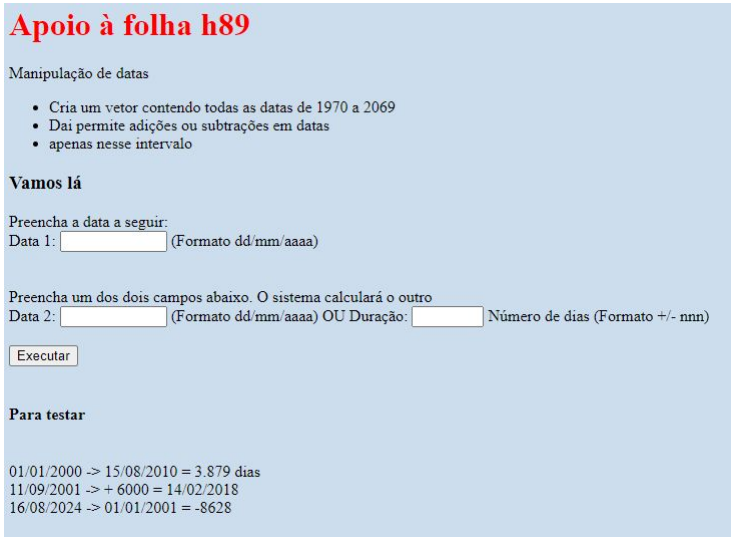
Note que a primeira coisa que o aplicativo pede é a digitação de uma data, em formato válido (`dd/mm/aaaa`) e compreendida no período válido (entre 1971 e 2069).

A seguir, o usuário pode escolher uma de duas opções

- Se optar por uma segunda data, o aplicativo vai calcular quantos dias separam as duas datas
- Se optar por uma quantidade (positiva ou negativa) de dias o sistema vai calcular a data obtida somando ou subtraindo a quantidade de dias informados em relação à primeira data informada.

A estratégia Devido à irregularidade do calendário, é muito difícil (e portanto sujeito a erros) a manipulação direta das variáveis dia, mês e ano em processos de adição e subtração de datas.

Analise a tela do seu aplicativo



Uma boa estratégia (certamente não a única) é criar um vetor de dias, onde o valor anterior corresponde à data de ontem, o atual é o dia de hoje e o próximo é amanha. Vamos chamar este vetor de vetor cronológico. Tendo construído este vetor (que é fixo e não precisa ser recalculado a cada momento), somar e subtrair datas se transforma no problema muito mais simples de somar e subtrair índices dentro desse vetor.

Guardar uma data em um inteiro Posto que uma data é um conjunto de 3 valores (dia, mês e ano, os 3 inteiros), para simplificar a construção do vetor cronológico, é conveniente combinar uma maneira de guardar os 3 valores em um único inteiro. Isto pode ser feito aplicando a seguinte expressão:

$$data = (dia \times 1000000) + (mes \times 10000) + ano$$

Para a construção do vetor cronológico é preciso estabelecer (combinar) com todos os que o usarão, qual o dia inicial do vetor (ou seja qual o valor de vetor[0]). Nada poderá ser feito com datas anteriores a esta. Nesta folha, vai-se usar como data inicial 1^o de janeiro de 1970, que vem a ser a data inicial de várias versões do sistema operacional Unix. Esta folha também vai realizar operações em 100 anos a partir dessa data, indo portanto até o ano de 2070. Diante disso, o tamanho do vetor cronológico é de mais ou menos 36600 elementos ($366 \times 100 = 36600$).

Próximo dia A função chave para construir o vetor cronológico é escrever uma função de nome amanhã, que receba uma data válida e devolva a data que corresponde a amanhã.

```
function amanha(hoje){
 dia=Math.floor(hoje/1000000)
 mes=Math.floor((hoje-(dia*1000000))/10000)
 ano=Math.floor(hoje-((dia*1000000)+(mes*10000)))
 if ((dia==31)&&(mes==12)){return 1010000+ano+1;}
 ud=0
 if ((dia==31)&&((mes==1)||(mes==3)||(mes==5)||
      (mes==7)||(mes==8)||(mes==10))){ud=1}
 if ((dia==30)&&((mes==4)||(mes==6)||(mes==9)||
      (mes==11))){ud=1}
 if ((ano%4==0)&&((ano%100!=0)||(ano%400==0))){
     bis=1}
 else {
     bis=0}
 if ((dia==28)&&(mes==2)&&(bis==0)){ud=1}
 if ((dia==29)&&(mes==2)&&(bis==1)){ud=1}
 if (ud==1){
    return 1000000+(10000*(mes+1))+ano
 }
 else {
    return (1000000*(dia+1))+(10000*mes)+ano
 }
}
```

Agora a criação e preenchimento do vetor cronológico

```
let vd = new Array(36600).fill(0)
function montavet(){
   vd[0]=01011970
   i=0
   while (i<36600){
      vd[i+1]=amanha(vd[i])
   i++
   }
}
```

A manipulação das datas no aplicativo é

```
function execu(){
   dt1=document.getElementById('dt1').value;
   dt2=document.getElementById('dt2').value;
   dur=document.getElementById('dur').value;
   d1=dt1.split('/')
   d1=d1.map(Number)
   x1=(d1[0]*1000000)+(d1[1]*10000)+d1[2]
   i1=vd.indexOf(x1)
   if (dur==""){
      d2=dt2.split('/')
      d2=d2.map(Number)
      x2=(d2[0]*1000000)+(d2[1]*10000)+d2[2]
   i2=vd.indexOf(x2)
   console.log('i2='+i2)
   du=i2-i1
   document.getElementById('dur').value=du
      }
 else {
      i2=i1+parseInt(dur)
      df=Math.floor(vd[i2]/1000000)
      mf=Math.floor((vd[i2]-(df*1000000))/10000)
      af=vd[i2]%10000
   document.getElementById('dt2').value=
           df+'/'+mf+'/'+af
   }
}
```

☞ Para você fazer

Escreva o aplicativo aqui descrito. Mostre para o professor. Atente que haverá solicitações *ad hoc* durante a avaliação. Para testar seu aplicativo, use os exemplos dados na folha, ou quaisquer outros.

Avaliação:

Depois que programar o aplicativo calcule quantos dias há entre 6/11/2000 e 6/ 3/2005 .

Dada a data 4/ 9/1992 qual a data obtida depois de 1195 dias?

Responda aqui:

Dias de diferença	Data achada

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 30/08/2024 - 11:20:57.0
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Operações com datas VIVXh89a V: 1.02
76032 FELIPE EIJI KANO
24FRO404 - 5 _____ / _____ / _____

Operações com datas

Mais um exercício para desenvolver as habilidades até agora trabalhadas, a saber: HTML, CSS e finalmente a programação Javascript. O aplicativo é para fazer cálculos envolvendo datas.

Mas para isso, vamos trabalhar alguns conceitos em JS, vejamos

Função de inicialização Se você precisar executar algo uma única vez ao carregar a página, um caminho possível é

- Definir `<body onload="inicializa()">` que informa que quando a página for carregada, deve ser executada a função `inicializa`.
- No script correspondente, defina a função `inicializa` fazendo o que deve ser feito.

Separar elementos de uma lista Supondo uma string com elementos separados por barras o comando `xx=lista.split('/')` vai criar uma lista onde cada elemento da string original será um elemento da lista

Converter lista de strings em lista de números Agora, o passo seguinte é converter cada elemento em um número. O comando é `yy=xx.map(Number)`. A função `map` aplica a função citada (neste caso é a `Number`) a cada elemento da lista.

Lista de saída No exercício desta folha você eventualmente precisa definir uma data de saída. A solução é

```
resp=' '
resp=dia+'/'+mes+"/"+ano
```

Note que o JS é suficientemente moderno para entender que valor é numérico mas como há strings na expressão (`"/"`) o sinal de adição não é adição e sim concatenação.

A encomenda

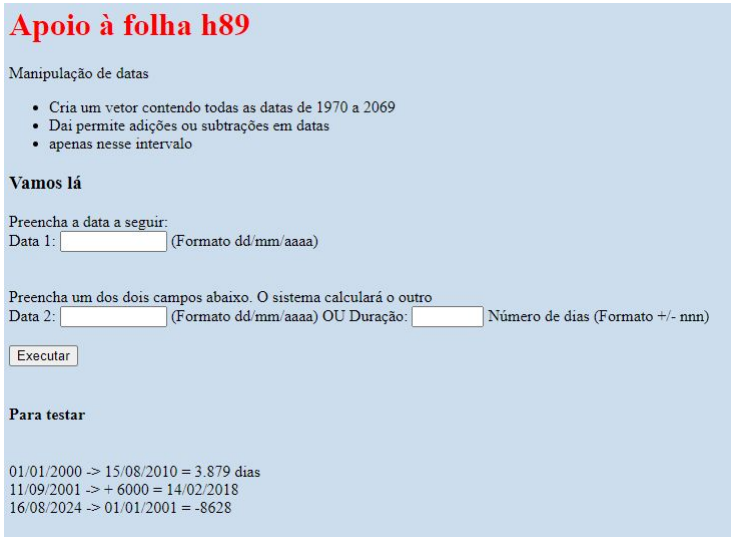
Note que a primeira coisa que o aplicativo pede é a digitação de uma data, em formato válido (`dd/mm/aaaa`) e compreendida no período válido (entre 1971 e 2069).

A seguir, o usuário pode escolher uma de duas opções

- Se optar por uma segunda data, o aplicativo vai calcular quantos dias separam as duas datas
- Se optar por uma quantidade (positiva ou negativa) de dias o sistema vai calcular a data obtida somando ou subtraindo a quantidade de dias informados em relação à primeira data informada.

A estratégia Devido à irregularidade do calendário, é muito difícil (e portanto sujeito a erros) a manipulação direta das variáveis dia, mês e ano em processos de adição e subtração de datas.

Analise a tela do seu aplicativo



Uma boa estratégia (certamente não a única) é criar um vetor de dias, onde o valor anterior corresponde à data de ontem, o atual é o dia de hoje e o próximo é amanha. Vamos chamar este vetor de vetor cronológico. Tendo construído este vetor (que é fixo e não precisa ser recalculado a cada momento), somar e subtrair datas se transforma no problema muito mais simples de somar e subtrair índices dentro desse vetor.

Guardar uma data em um inteiro Posto que uma data é um conjunto de 3 valores (dia, mês e ano, os 3 inteiros), para simplificar a construção do vetor cronológico, é conveniente combinar uma maneira de guardar os 3 valores em um único inteiro. Isto pode ser feito aplicando a seguinte expressão:

$$data = (dia \times 1000000) + (mes \times 10000) + ano$$

Para a construção do vetor cronológico é preciso estabelecer (combinar) com todos os que o usarão, qual o dia inicial do vetor (ou seja qual o valor de vetor[0]). Nada poderá ser feito com datas anteriores a esta. Nesta folha, vai-se usar como data inicial 1^o de janeiro de 1970, que vem a ser a data inicial de várias versões do sistema operacional Unix. Esta folha também vai realizar operações em 100 anos a partir dessa data, indo portanto até o ano de 2070. Diante disso, o tamanho do vetor cronológico é de mais ou menos 36600 elementos ($366 \times 100 = 36600$).

Próximo dia A função chave para construir o vetor cronológico é escrever uma função de nome amanhã, que receba uma data válida e devolva a data que corresponde a amanhã.

```
function amanha(hoje){
 dia=Math.floor(hoje/1000000)
 mes=Math.floor((hoje-(dia*1000000))/10000)
 ano=Math.floor(hoje-((dia*1000000)+(mes*10000)))
 if ((dia==31)&&(mes==12)){return 1010000+ano+1;}
 ud=0
 if ((dia==31)&&((mes==1)||(mes==3)||(mes==5)||
      (mes==7)||(mes==8)||(mes==10))){ud=1}
 if ((dia==30)&&((mes==4)||(mes==6)||(mes==9)||
      (mes==11))){ud=1}
 if ((ano%4==0)&&((ano%100!=0)||(ano%400==0))){
     bis=1}
 else {
     bis=0}
 if ((dia==28)&&(mes==2)&&(bis==0)){ud=1}
 if ((dia==29)&&(mes==2)&&(bis==1)){ud=1}
 if (ud==1){
    return 1000000+(10000*(mes+1))+ano
 }
 else {
    return (1000000*(dia+1))+(10000*mes)+ano
 }
}
```

Agora a criação e preenchimento do vetor cronológico

```
let vd = new Array(36600).fill(0)
function montavet(){
   vd[0]=01011970
   i=0
   while (i<36600){
      vd[i+1]=amanha(vd[i])
   i++
   }
}
```

A manipulação das datas no aplicativo é

```
function execu(){
   dt1=document.getElementById('dt1').value;
   dt2=document.getElementById('dt2').value;
   dur=document.getElementById('dur').value;
   d1=dt1.split('/')
   d1=d1.map(Number)
   x1=(d1[0]*1000000)+(d1[1]*10000)+d1[2]
   i1=vd.indexOf(x1)
   if (dur==""){
      d2=dt2.split('/')
      d2=d2.map(Number)
      x2=(d2[0]*1000000)+(d2[1]*10000)+d2[2]
   i2=vd.indexOf(x2)
   console.log('i2='+i2)
   du=i2-i1
   document.getElementById('dur').value=du
      }
 else {
      i2=i1+parseInt(dur)
      df=Math.floor(vd[i2]/1000000)
      mf=Math.floor((vd[i2]-(df*1000000))/10000)
      af=vd[i2]%10000
   document.getElementById('dt2').value=
           df+'/'+mf+'/'+af
   }
}
```

☞ Para você fazer

Escreva o aplicativo aqui descrito. Mostre para o professor. Atente que haverá solicitações *ad hoc* durante a avaliação. Para testar seu aplicativo, use os exemplos dados na folha, ou quaisquer outros.

Avaliação:

Depois que programar o aplicativo calcule quantos dias há entre 4/ 3/2002 e 5/ 2/2005 .

Dada a data 18/ 2/1989 qual a data obtida depois de 1804 dias?

Responda aqui:

Dias de diferença	Data achada

404-76032 - /

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 30/08/2024 - 11:20:57.0
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Operações com datas VIVXh89a V: 1.02
76049 FELIPE JESMIEL LEITE
24FRO404 - 6 _____ / _____ / _____

Operações com datas

Mais um exercício para desenvolver as habilidades até agora trabalhadas, a saber: HTML, CSS e finalmente a programação Javascript. O aplicativo é para fazer cálculos envolvendo datas.

Mas para isso, vamos trabalhar alguns conceitos em JS, vejamos

Função de inicialização Se você precisar executar algo uma única vez ao carregar a página, um caminho possível é

- Definir `<body onload="inicializa()">` que informa que quando a página for carregada, deve ser executada a função `inicializa`.
- No script correspondente, defina a função `inicializa` fazendo o que deve ser feito.

Separar elementos de uma lista Supondo uma string com elementos separados por barras o comando `xx=lista.split('/')` vai criar uma lista onde cada elemento da string original será um elemento da lista

Converter lista de strings em lista de números Agora, o passo seguinte é converter cada elemento em um número. O comando é `yy=xx.map(Number)`. A função `map` aplica a função citada (neste caso é a `Number`) a cada elemento da lista.

Lista de saída No exercício desta folha você eventualmente precisa definir uma data de saída. A solução é

```
resp=' '
resp=dia+'/'+mes+"/"+ano
```

Note que o JS é suficientemente moderno para entender que valor é numérico mas como há strings na expressão (`"/"`) o sinal de adição não é adição e sim concatenação.

A encomenda

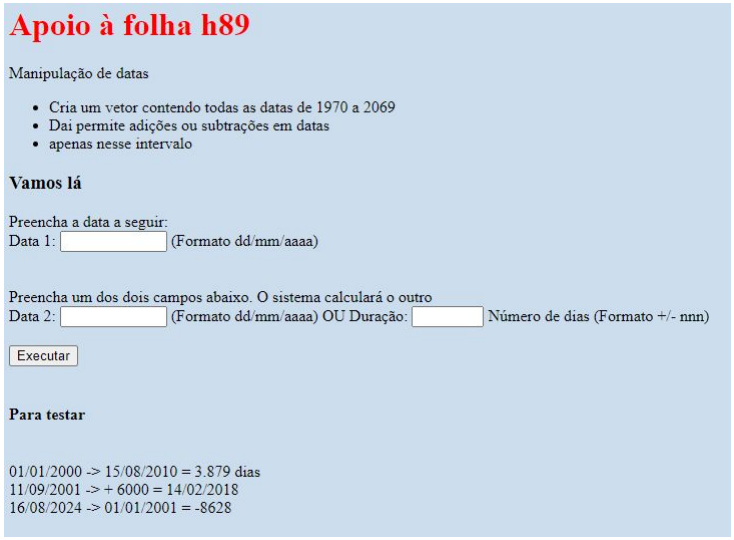
Note que a primeira coisa que o aplicativo pede é a digitação de uma data, em formato válido (`dd/mm/aaaa`) e compreendida no período válido (entre 1971 e 2069).

A seguir, o usuário pode escolher uma de duas opções

- Se optar por uma segunda data, o aplicativo vai calcular quantos dias separam as duas datas
- Se optar por uma quantidade (positiva ou negativa) de dias o sistema vai calcular a data obtida somando ou subtraindo a quantidade de dias informados em relação à primeira data informada.

A estratégia Devido à irregularidade do calendário, é muito difícil (e portanto sujeito a erros) a manipulação direta das variáveis dia, mês e ano em processos de adição e subtração de datas.

Analise a tela do seu aplicativo



Uma boa estratégia (certamente não a única) é criar um vetor de dias, onde o valor anterior corresponde à data de ontem, o atual é o dia de hoje e o próximo é amanha. Vamos chamar este vetor de vetor cronológico. Tendo construído este vetor (que é fixo e não precisa ser recalculado a cada momento), somar e subtrair datas se transforma no problema muito mais simples de somar e subtrair índices dentro desse vetor.

Guardar uma data em um inteiro Posto que uma data é um conjunto de 3 valores (dia, mês e ano, os 3 inteiros), para simplificar a construção do vetor cronológico, é conveniente combinar uma maneira de guardar os 3 valores em um único inteiro. Isto pode ser feito aplicando a seguinte expressão:

$$data = (dia \times 1000000) + (mes \times 10000) + ano$$

Para a construção do vetor cronológico é preciso estabelecer (combinar) com todos os que o usarão, qual o dia inicial do vetor (ou seja qual o valor de vetor[0]). Nada poderá ser feito com datas anteriores a esta. Nesta folha, vai-se usar como data inicial 1^o de janeiro de 1970, que vem a ser a data inicial de várias versões do sistema operacional Unix. Esta folha também vai realizar operações em 100 anos a partir dessa data, indo portanto até o ano de 2070. Diante disso, o tamanho do vetor cronológico é de mais ou menos 36600 elementos ($366 \times 100 = 36600$).

Próximo dia A função chave para construir o vetor cronológico é escrever uma função de nome amanhã, que receba uma data válida e devolva a data que corresponde a amanhã.

```
function amanha(hoje){
 dia=Math.floor(hoje/1000000)
 mes=Math.floor((hoje-(dia*1000000))/10000)
 ano=Math.floor(hoje-((dia*1000000)+(mes*10000)))
 if ((dia==31)&&(mes==12)){return 1010000+ano+1;}
 ud=0
 if ((dia==31)&&((mes==1)||(mes==3)||(mes==5)||
      (mes==7)||(mes==8)||(mes==10))){ud=1}
 if ((dia==30)&&((mes==4)||(mes==6)||(mes==9)||
      (mes==11))){ud=1}
 if ((ano%4==0)&&((ano%100!=0)||(ano%400==0))){
     bis=1}
 else {
     bis=0}
 if ((dia==28)&&(mes==2)&&(bis==0)){ud=1}
 if ((dia==29)&&(mes==2)&&(bis==1)){ud=1}
 if (ud==1){
    return 1000000+(10000*(mes+1))+ano
 }
 else {
    return (1000000*(dia+1))+(10000*mes)+ano
 }
}
```

Agora a criação e preenchimento do vetor cronológico

```
let vd = new Array(36600).fill(0)
function montavet(){
   vd[0]=01011970
   i=0
   while (i<36600){
      vd[i+1]=amanha(vd[i])
   i++
   }
}
```

A manipulação das datas no aplicativo é

```
function execu(){
   dt1=document.getElementById('dt1').value;
   dt2=document.getElementById('dt2').value;
   dur=document.getElementById('dur').value;
   d1=dt1.split('/')
   d1=d1.map(Number)
   x1=(d1[0]*1000000)+(d1[1]*10000)+d1[2]
   i1=vd.indexOf(x1)
   if (dur==""){
      d2=dt2.split('/')
      d2=d2.map(Number)
      x2=(d2[0]*1000000)+(d2[1]*10000)+d2[2]
   i2=vd.indexOf(x2)
   console.log('i2='+i2)
   du=i2-i1
   document.getElementById('dur').value=du
      }
 else {
      i2=i1+parseInt(dur)
      df=Math.floor(vd[i2]/1000000)
      mf=Math.floor((vd[i2]-(df*1000000))/10000)
      af=vd[i2]%10000
   document.getElementById('dt2').value=
          df+'/'+mf+'/'+af
   }
}
```

☞ Para você fazer

Escreva o aplicativo aqui descrito. Mostre para o professor. Atente que haverá solicitações *ad hoc* durante a avaliação. Para testar seu aplicativo, use os exemplos dados na folha, ou quaisquer outros.

Avaliação:

Depois que programar o aplicativo calcule quantos dias há entre 19/ 2/1997 e 4/11/2003 .

Dada a data 29/ 5/1981 qual a data obtida depois de 1686 dias?

Responda aqui:

Dias de diferença	Data achada

404-76049 - /

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 30/08/2024 - 11:20:57.0
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Operações com datas VIVXh89a V: 1.02
76056 GABRIELA BERNARDO PASQUALOTT
24FRO404 - 7 _____ / _____ / _____

Operações com datas

Mais um exercício para desenvolver as habilidades até agora trabalhadas, a saber: HTML, CSS e finalmente a programação Javascript. O aplicativo é para fazer cálculos envolvendo datas.

Mas para isso, vamos trabalhar alguns conceitos em JS, vejamos

Função de inicialização Se você precisar executar algo uma única vez ao carregar a página, um caminho possível é

- Definir `<body onload="inicializa()">` que informa que quando a página for carregada, deve ser executada a função `inicializa`.
- No script correspondente, defina a função `inicializa` fazendo o que deve ser feito.

Separar elementos de uma lista Supondo uma string com elementos separados por barras o comando `xx=lista.split('/')` vai criar uma lista onde cada elemento da string original será um elemento da lista

Converter lista de strings em lista de números Agora, o passo seguinte é converter cada elemento em um número. O comando é `yy=xx.map(Number)`. A função `map` aplica a função citada (neste caso é a `Number`) a cada elemento da lista.

Lista de saída No exercício desta folha você eventualmente precisa definir uma data de saída. A solução é

```
resp=' '
resp=dia+'/'+mes+"/"+ano
```

Note que o JS é suficientemente moderno para entender que valor é numérico mas como há strings na expressão (`"/"`) o sinal de adição não é adição e sim concatenação.

A encomenda

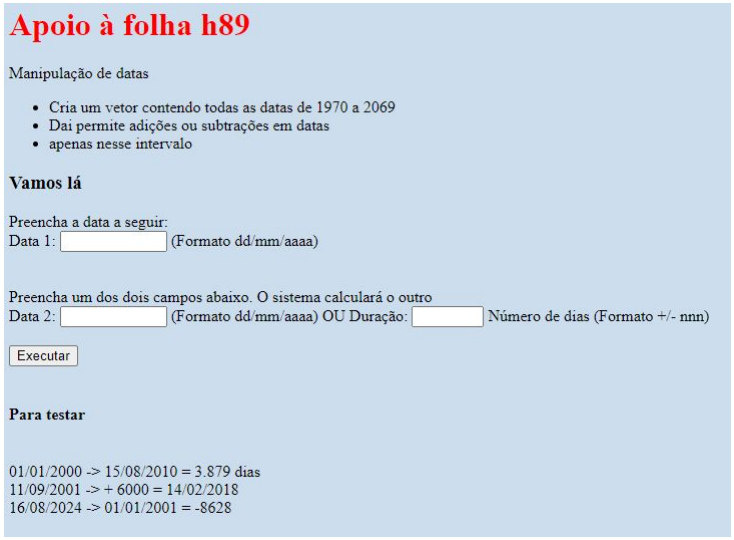
Note que a primeira coisa que o aplicativo pede é a digitação de uma data, em formato válido (`dd/mm/aaaa`) e compreendida no período válido (entre 1971 e 2069).

A seguir, o usuário pode escolher uma de duas opções

- Se optar por uma segunda data, o aplicativo vai calcular quantos dias separam as duas datas
- Se optar por uma quantidade (positiva ou negativa) de dias o sistema vai calcular a data obtida somando ou subtraindo a quantidade de dias informados em relação à primeira data informada.

A estratégia Devido à irregularidade do calendário, é muito difícil (e portanto sujeito a erros) a manipulação direta das variáveis dia, mês e ano em processos de adição e subtração de datas.

Analise a tela do seu aplicativo



Uma boa estratégia (certamente não a única) é criar um vetor de dias, onde o valor anterior corresponde à data de ontem, o atual é o dia de hoje e o próximo é amanha. Vamos chamar este vetor de vetor cronológico. Tendo construído este vetor (que é fixo e não precisa ser recalculado a cada momento), somar e subtrair datas se transforma no problema muito mais simples de somar e subtrair índices dentro desse vetor.

Guardar uma data em um inteiro Posto que uma data é um conjunto de 3 valores (dia, mês e ano, os 3 inteiros), para simplificar a construção do vetor cronológico, é conveniente combinar uma maneira de guardar os 3 valores em um único inteiro. Isto pode ser feito aplicando a seguinte expressão:

$$data = (dia \times 1000000) + (mes \times 10000) + ano$$

Para a construção do vetor cronológico é preciso estabelecer (combinar) com todos os que o usarão, qual o dia inicial do vetor (ou seja qual o valor de vetor[0]). Nada poderá ser feito com datas anteriores a esta. Nesta folha, vai-se usar como data inicial 1^o de janeiro de 1970, que vem a ser a data inicial de várias versões do sistema operacional Unix. Esta folha também vai realizar operações em 100 anos a partir dessa data, indo portanto até o ano de 2070. Diante disso, o tamanho do vetor cronológico é de mais ou menos 36600 elementos ($366 \times 100 = 36600$).

Próximo dia A função chave para construir o vetor cronológico é escrever uma função de nome amanhã, que receba uma data válida e devolva a data que corresponde a amanhã.

```
function amanha(hoje){
 dia=Math.floor(hoje/1000000)
 mes=Math.floor((hoje-(dia*1000000))/10000)
 ano=Math.floor(hoje-((dia*1000000)+(mes*10000)))
 if ((dia==31)&&(mes==12)){return 1010000+ano+1;}
 ud=0
 if ((dia==31)&&((mes==1)||(mes==3)||(mes==5)||
      (mes==7)||(mes==8)||(mes==10))){ud=1}
 if ((dia==30)&&((mes==4)||(mes==6)||(mes==9)||
      (mes==11))){ud=1}
 if ((ano%4==0)&&((ano%100!=0)||(ano%400==0))){
     bis=1}
 else {
     bis=0}
 if ((dia==28)&&(mes==2)&&(bis==0)){ud=1}
 if ((dia==29)&&(mes==2)&&(bis==1)){ud=1}
 if (ud==1){
    return 1000000+(10000*(mes+1))+ano
 }
 else {
    return (1000000*(dia+1))+(10000*mes)+ano
 }
}
```

Agora a criação e preenchimento do vetor cronológico

```
let vd = new Array(36600).fill(0)
function montavet(){
   vd[0]=01011970
   i=0
   while (i<36600){
      vd[i+1]=amanha(vd[i])
   i++
   }
}
```

A manipulação das datas no aplicativo é

```
function execu(){
   dt1=document.getElementById('dt1').value;
   dt2=document.getElementById('dt2').value;
   dur=document.getElementById('dur').value;
   d1=dt1.split('/')
   d1=d1.map(Number)
   x1=(d1[0]*1000000)+(d1[1]*10000)+d1[2]
   i1=vd.indexOf(x1)
   if (dur==""){
      d2=dt2.split('/')
      d2=d2.map(Number)
      x2=(d2[0]*1000000)+(d2[1]*10000)+d2[2]
   i2=vd.indexOf(x2)
   console.log('i2='+i2)
   du=i2-i1
   document.getElementById('dur').value=du
      }
 else {
      i2=i1+parseInt(dur)
      df=Math.floor(vd[i2]/1000000)
      mf=Math.floor((vd[i2]-(df*1000000))/10000)
      af=vd[i2]%10000
   document.getElementById('dt2').value=
           df+'/'+mf+'/'+af
   }
}
```

☞ Para você fazer

Escreva o aplicativo aqui descrito. Mostre para o professor. Atente que haverá solicitações *ad hoc* durante a avaliação. Para testar seu aplicativo, use os exemplos dados na folha, ou quaisquer outros.

Avaliação:

Depois que programar o aplicativo calcule quantos dias há entre 30/ 8/1994 e 26/ 7/1997 .

Dada a data 22/11/1976 qual a data obtida depois de 1736 dias?

Responda aqui:

Dias de diferença	Data achada

404-76056 - /

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 30/08/2024 - 11:20:57.0
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Operações com datas VIVXh89a V: 1.02
76063 GUILHERME VITOR DA SILVA
24FRO404 - 8 _____ / _____ / _____

Operações com datas

Mais um exercício para desenvolver as habilidades até agora trabalhadas, a saber: HTML, CSS e finalmente a programação Javascript. O aplicativo é para fazer cálculos envolvendo datas.

Mas para isso, vamos trabalhar alguns conceitos em JS, vejamos

Função de inicialização Se você precisar executar algo uma única vez ao carregar a página, um caminho possível é

- Definir `<body onload="inicializa()">` que informa que quando a página for carregada, deve ser executada a função `inicializa`.
- No script correspondente, defina a função `inicializa` fazendo o que deve ser feito.

Separar elementos de uma lista Supondo uma string com elementos separados por barras o comando `xx=lista.split('/')` vai criar uma lista onde cada elemento da string original será um elemento da lista

Converter lista de strings em lista de números Agora, o passo seguinte é converter cada elemento em um número. O comando é `yy=xx.map(Number)`. A função `map` aplica a função citada (neste caso é a `Number`) a cada elemento da lista.

Lista de saída No exercício desta folha você eventualmente precisa definir uma data de saída. A solução é

```
resp=' '
resp=dia+'/'+mes+"/"+ano
```

Note que o JS é suficientemente moderno para entender que valor é numérico mas como há strings na expressão (`"/"`) o sinal de adição não é adição e sim concatenação.

A encomenda

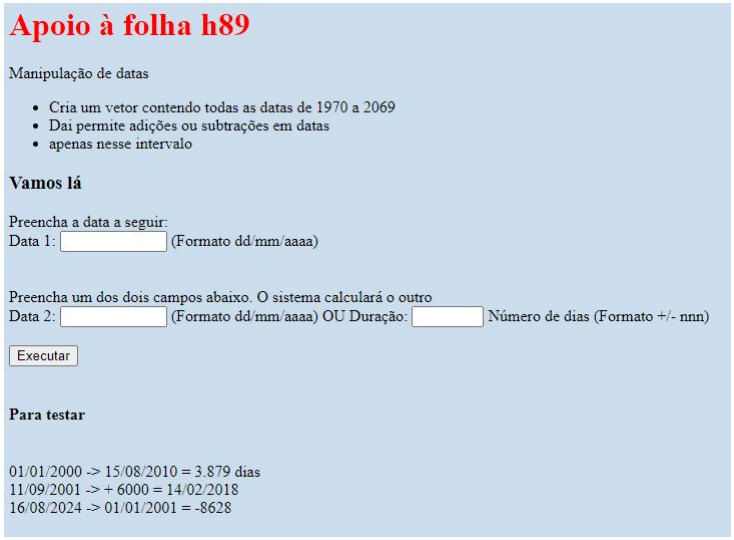
Note que a primeira coisa que o aplicativo pede é a digitação de uma data, em formato válido (`dd/mm/aaaa`) e compreendida no período válido (entre 1971 e 2069).

A seguir, o usuário pode escolher uma de duas opções

- Se optar por uma segunda data, o aplicativo vai calcular quantos dias separam as duas datas
- Se optar por uma quantidade (positiva ou negativa) de dias o sistema vai calcular a data obtida somando ou subtraindo a quantidade de dias informados em relação à primeira data informada.

A estratégia Devido à irregularidade do calendário, é muito difícil (e portanto sujeito a erros) a manipulação direta das variáveis dia, mês e ano em processos de adição e subtração de datas.

Analise a tela do seu aplicativo



Uma boa estratégia (certamente não a única) é criar um vetor de dias, onde o valor anterior corresponde à data de ontem, o atual é o dia de hoje e o próximo é amanha. Vamos chamar este vetor de vetor cronológico. Tendo construído este vetor (que é fixo e não precisa ser recalculado a cada momento), somar e subtrair datas se transforma no problema muito mais simples de somar e subtrair índices dentro desse vetor.

Guardar uma data em um inteiro Posto que uma data é um conjunto de 3 valores (dia, mês e ano, os 3 inteiros), para simplificar a construção do vetor cronológico, é conveniente combinar uma maneira de guardar os 3 valores em um único inteiro. Isto pode ser feito aplicando a seguinte expressão:

$$data = (dia \times 1000000) + (mes \times 10000) + ano$$

Para a construção do vetor cronológico é preciso estabelecer (combinar) com todos os que o usarão, qual o dia inicial do vetor (ou seja qual o valor de vetor[0]). Nada poderá ser feito com datas anteriores a esta. Nesta folha, vai-se usar como data inicial 1^o de janeiro de 1970, que vem a ser a data inicial de várias versões do sistema operacional Unix. Esta folha também vai realizar operações em 100 anos a partir dessa data, indo portanto até o ano de 2070. Diante disso, o tamanho do vetor cronológico é de mais ou menos 36600 elementos ($366 \times 100 = 36600$).

Próximo dia A função chave para construir o vetor cronológico é escrever uma função de nome amanhã, que receba uma data válida e devolva a data que corresponde a amanhã.

```
function amanha(hoje){
 dia=Math.floor(hoje/1000000)
 mes=Math.floor((hoje-(dia*1000000))/10000)
 ano=Math.floor(hoje-((dia*1000000)+(mes*10000)))
 if ((dia==31)&&(mes==12)){return 1010000+ano+1;}
 ud=0
 if ((dia==31)&&((mes==1)||(mes==3)||(mes==5)||
      (mes==7)||(mes==8)||(mes==10))){ud=1}
 if ((dia==30)&&((mes==4)||(mes==6)||(mes==9)||
      (mes==11))){ud=1}
 if ((ano%4==0)&&((ano%100!=0)||(ano%400==0))){
     bis=1}
 else {
     bis=0}
 if ((dia==28)&&(mes==2)&&(bis==0)){ud=1}
 if ((dia==29)&&(mes==2)&&(bis==1)){ud=1}
 if (ud==1){
    return 1000000+(10000*(mes+1))+ano
 }
 else {
    return (1000000*(dia+1))+(10000*mes)+ano
 }
}
```

Agora a criação e preenchimento do vetor cronológico

```
let vd = new Array(36600).fill(0)
function montavet(){
   vd[0]=01011970
   i=0
   while (i<36600){
      vd[i+1]=amanha(vd[i])
   i++
   }
}
```

A manipulação das datas no aplicativo é

```
function execu(){
   dt1=document.getElementById('dt1').value;
   dt2=document.getElementById('dt2').value;
   dur=document.getElementById('dur').value;
   d1=dt1.split('/')
   d1=d1.map(Number)
   x1=(d1[0]*1000000)+(d1[1]*10000)+d1[2]
   i1=vd.indexOf(x1)
   if (dur==""){
      d2=dt2.split('/')
      d2=d2.map(Number)
      x2=(d2[0]*1000000)+(d2[1]*10000)+d2[2]
   i2=vd.indexOf(x2)
   console.log('i2='+i2)
   du=i2-i1
   document.getElementById('dur').value=du
      }
 else {
      i2=i1+parseInt(dur)
      df=Math.floor(vd[i2]/1000000)
      mf=Math.floor((vd[i2]-(df*1000000))/10000)
      af=vd[i2]%10000
   document.getElementById('dt2').value=
          df+'/'+mf+'/'+af
   }
}
```

☞ Para você fazer

Escreva o aplicativo aqui descrito. Mostre para o professor. Atente que haverá solicitações *ad hoc* durante a avaliação. Para testar seu aplicativo, use os exemplos dados na folha, ou quaisquer outros.
Avaliação:

Depois que programar o aplicativo calcule quantos dias há entre 18/ 4/1982 e 27/11/1987 .

Dada a data 14/ 1/1982 qual a data obtida depois de 1470 dias?

Responda aqui:

Dias de diferença	Data achada

404-76063 - /

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 30/08/2024 - 11:20:57.0
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Operações com datas VIVXh89a V: 1.02
76087 LAURA PACHECO DELATTRE
24FRO404 - 9 _____ / _____ / _____

Operações com datas

Mais um exercício para desenvolver as habilidades até agora trabalhadas, a saber: HTML, CSS e finalmente a programação Javascript. O aplicativo é para fazer cálculos envolvendo datas.

Mas para isso, vamos trabalhar alguns conceitos em JS, vejamos

Função de inicialização Se você precisar executar algo uma única vez ao carregar a página, um caminho possível é

- Definir `<body onload="inicializa()">` que informa que quando a página for carregada, deve ser executada a função `inicializa`.
- No script correspondente, defina a função `inicializa` fazendo o que deve ser feito.

Separar elementos de uma lista Supondo uma string com elementos separados por barras o comando `xx=lista.split('/')` vai criar uma lista onde cada elemento da string original será um elemento da lista

Converter lista de strings em lista de números Agora, o passo seguinte é converter cada elemento em um número. O comando é `yy=xx.map(Number)`. A função `map` aplica a função citada (neste caso é a `Number`) a cada elemento da lista.

Lista de saída No exercício desta folha você eventualmente precisa definir uma data de saída. A solução é

```
resp=' '
resp=dia+'/'+mes+"/"+ano
```

Note que o JS é suficientemente moderno para entender que valor é numérico mas como há strings na expressão (`"/"`) o sinal de adição não é adição e sim concatenação.

A encomenda

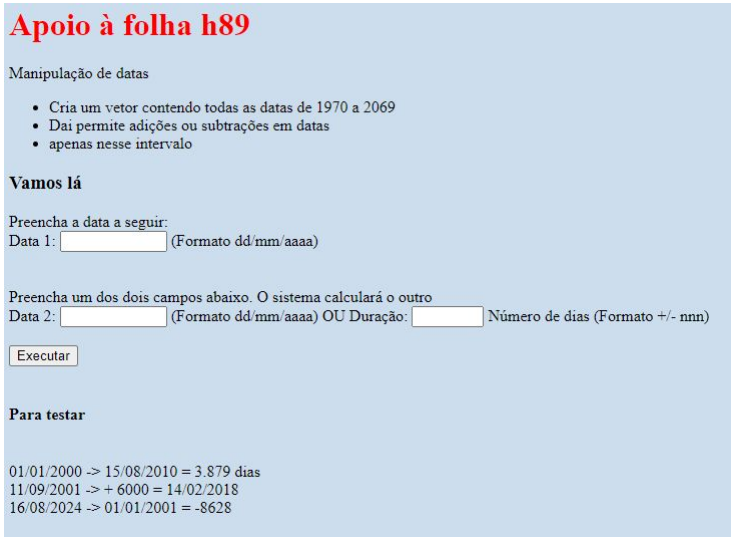
Note que a primeira coisa que o aplicativo pede é a digitação de uma data, em formato válido (`dd/mm/aaaa`) e compreendida no período válido (entre 1971 e 2069).

A seguir, o usuário pode escolher uma de duas opções

- Se optar por uma segunda data, o aplicativo vai calcular quantos dias separam as duas datas
- Se optar por uma quantidade (positiva ou negativa) de dias o sistema vai calcular a data obtida somando ou subtraindo a quantidade de dias informados em relação à primeira data informada.

A estratégia Devido à irregularidade do calendário, é muito difícil (e portanto sujeito a erros) a manipulação direta das variáveis dia, mês e ano em processos de adição e subtração de datas.

Analise a tela do seu aplicativo



Uma boa estratégia (certamente não a única) é criar um vetor de dias, onde o valor anterior corresponde à data de ontem, o atual é o dia de hoje e o próximo é amanha. Vamos chamar este vetor de vetor cronológico. Tendo construído este vetor (que é fixo e não precisa ser recalculado a cada momento), somar e subtrair datas se transforma no problema muito mais simples de somar e subtrair índices dentro desse vetor.

Guardar uma data em um inteiro Posto que uma data é um conjunto de 3 valores (dia, mês e ano, os 3 inteiros), para simplificar a construção do vetor cronológico, é conveniente combinar uma maneira de guardar os 3 valores em um único inteiro. Isto pode ser feito aplicando a seguinte expressão:

$$data = (dia \times 1000000) + (mes \times 10000) + ano$$

Para a construção do vetor cronológico é preciso estabelecer (combinar) com todos os que o usarão, qual o dia inicial do vetor (ou seja qual o valor de vetor[0]). Nada poderá ser feito com datas anteriores a esta. Nesta folha, vai-se usar como data inicial 1^o de janeiro de 1970, que vem a ser a data inicial de várias versões do sistema operacional Unix. Esta folha também vai realizar operações em 100 anos a partir dessa data, indo portanto até o ano de 2070. Diante disso, o tamanho do vetor cronológico é de mais ou menos 36600 elementos ($366 \times 100 = 36600$).

Próximo dia A função chave para construir o vetor cronológico é escrever uma função de nome amanhã, que receba uma data válida e devolva a data que corresponde a amanhã.

```
function amanha(hoje){
 dia=Math.floor(hoje/1000000)
 mes=Math.floor((hoje-(dia*1000000))/10000)
 ano=Math.floor(hoje-((dia*1000000)+(mes*10000)))
 if ((dia==31)&&(mes==12)){return 1010000+ano+1;}
 ud=0
 if ((dia==31)&&((mes==1)||(mes==3)||(mes==5)||
      (mes==7)||(mes==8)||(mes==10))){ud=1}
 if ((dia==30)&&((mes==4)||(mes==6)||(mes==9)||
      (mes==11))){ud=1}
 if ((ano%4==0)&&((ano%100!=0)||(ano%400==0))){
     bis=1}
 else {
     bis=0}
 if ((dia==28)&&(mes==2)&&(bis==0)){ud=1}
 if ((dia==29)&&(mes==2)&&(bis==1)){ud=1}
 if (ud==1){
    return 1000000+(10000*(mes+1))+ano
 }
 else {
    return (1000000*(dia+1))+(10000*mes)+ano
 }
}
```

Agora a criação e preenchimento do vetor cronológico

```
let vd = new Array(36600).fill(0)
function montavet(){
   vd[0]=01011970
   i=0
   while (i<36600){
      vd[i+1]=amanha(vd[i])
   i++
   }
}
```

A manipulação das datas no aplicativo é

```
function execu(){
   dt1=document.getElementById('dt1').value;
   dt2=document.getElementById('dt2').value;
   dur=document.getElementById('dur').value;
   d1=dt1.split('/')
   d1=d1.map(Number)
   x1=(d1[0]*1000000)+(d1[1]*10000)+d1[2]
   i1=vd.indexOf(x1)
   if (dur==""){
      d2=dt2.split('/')
      d2=d2.map(Number)
      x2=(d2[0]*1000000)+(d2[1]*10000)+d2[2]
   i2=vd.indexOf(x2)
   console.log('i2='+i2)
   du=i2-i1
   document.getElementById('dur').value=du
      }
 else {
      i2=i1+parseInt(dur)
      df=Math.floor(vd[i2]/1000000)
      mf=Math.floor((vd[i2]-(df*1000000))/10000)
      af=vd[i2]%10000
   document.getElementById('dt2').value=
          df+'/'+mf+'/'+af
   }
}
```

☞ Para você fazer

Escreva o aplicativo aqui descrito. Mostre para o professor. Atente que haverá solicitações *ad hoc* durante a avaliação. Para testar seu aplicativo, use os exemplos dados na folha, ou quaisquer outros.

Avaliação:

Depois que programar o aplicativo calcule quantos dias há entre 27/ 2/1979 e 9/ 3/1986 .

Dada a data 24/ 9/1992 qual a data obtida depois de 1045 dias?

Responda aqui:

Dias de diferença	Data achada

404-76087 - /

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 30/08/2024 - 11:20:57.0
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Operações com datas VIVXh89a V: 1.02
76687 LAURO HENRIK RUTZ
24FRO404 - 10 _____ / _____ / _____

Operações com datas

Mais um exercício para desenvolver as habilidades até agora trabalhadas, a saber: HTML, CSS e finalmente a programação Javascript. O aplicativo é para fazer cálculos envolvendo datas.

Mas para isso, vamos trabalhar alguns conceitos em JS, vejamos

Função de inicialização Se você precisar executar algo uma única vez ao carregar a página, um caminho possível é

- Definir `<body onload="inicializa()">` que informa que quando a página for carregada, deve ser executada a função `inicializa`.
- No script correspondente, defina a função `inicializa` fazendo o que deve ser feito.

Separar elementos de uma lista Supondo uma string com elementos separados por barras o comando `xx=lista.split('/')` vai criar uma lista onde cada elemento da string original será um elemento da lista

Converter lista de strings em lista de números Agora, o passo seguinte é converter cada elemento em um número. O comando é `yy=xx.map(Number)`. A função `map` aplica a função citada (neste caso é a `Number`) a cada elemento da lista.

Lista de saída No exercício desta folha você eventualmente precisa definir uma data de saída. A solução é

```
resp=' '
resp=dia+'/'+mes+"/"+ano
```

Note que o JS é suficientemente moderno para entender que valor é numérico mas como há strings na expressão (`"/"`) o sinal de adição não é adição e sim concatenação.

A encomenda

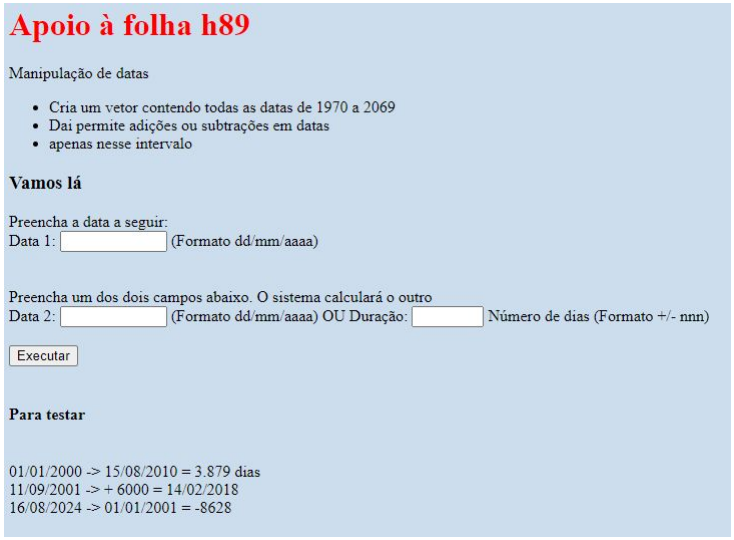
Note que a primeira coisa que o aplicativo pede é a digitação de uma data, em formato válido (`dd/mm/aaaa`) e compreendida no período válido (entre 1971 e 2069).

A seguir, o usuário pode escolher uma de duas opções

- Se optar por uma segunda data, o aplicativo vai calcular quantos dias separam as duas datas
- Se optar por uma quantidade (positiva ou negativa) de dias o sistema vai calcular a data obtida somando ou subtraindo a quantidade de dias informados em relação à primeira data informada.

A estratégia Devido à irregularidade do calendário, é muito difícil (e portanto sujeito a erros) a manipulação direta das variáveis dia, mês e ano em processos de adição e subtração de datas.

Analise a tela do seu aplicativo



Uma boa estratégia (certamente não a única) é criar um vetor de dias, onde o valor anterior corresponde à data de ontem, o atual é o dia de hoje e o próximo é amanha. Vamos chamar este vetor de vetor cronológico. Tendo construído este vetor (que é fixo e não precisa ser recalculado a cada momento), somar e subtrair datas se transforma no problema muito mais simples de somar e subtrair índices dentro desse vetor.

Guardar uma data em um inteiro Posto que uma data é um conjunto de 3 valores (dia, mês e ano, os 3 inteiros), para simplificar a construção do vetor cronológico, é conveniente combinar uma maneira de guardar os 3 valores em um único inteiro. Isto pode ser feito aplicando a seguinte expressão:

$$data = (dia \times 1000000) + (mes \times 10000) + ano$$

Para a construção do vetor cronológico é preciso estabelecer (combinar) com todos os que o usarão, qual o dia inicial do vetor (ou seja qual o valor de vetor[0]). Nada poderá ser feito com datas anteriores a esta. Nesta folha, vai-se usar como data inicial 1^o de janeiro de 1970, que vem a ser a data inicial de várias versões do sistema operacional Unix. Esta folha também vai realizar operações em 100 anos a partir dessa data, indo portanto até o ano de 2070. Diante disso, o tamanho do vetor cronológico é de mais ou menos 36600 elementos ($366 \times 100 = 36600$).

Próximo dia A função chave para construir o vetor cronológico é escrever uma função de nome amanhã, que receba uma data válida e devolva a data que corresponde a amanhã.

```
function amanha(hoje){
 dia=Math.floor(hoje/1000000)
 mes=Math.floor((hoje-(dia*1000000))/10000)
 ano=Math.floor(hoje-((dia*1000000)+(mes*10000)))
 if ((dia==31)&&(mes==12)){return 1010000+ano+1;}
 ud=0
 if ((dia==31)&&((mes==1)||(mes==3)||(mes==5)||
      (mes==7)||(mes==8)||(mes==10))){ud=1}
 if ((dia==30)&&((mes==4)||(mes==6)||(mes==9)||
      (mes==11))){ud=1}
 if ((ano%4==0)&&((ano%100!=0)||(ano%400==0))){
     bis=1}
 else {
     bis=0}
 if ((dia==28)&&(mes==2)&&(bis==0)){ud=1}
 if ((dia==29)&&(mes==2)&&(bis==1)){ud=1}
 if (ud==1){
    return 1000000+(10000*(mes+1))+ano
 }
 else {
    return (1000000*(dia+1))+(10000*mes)+ano
 }
}
```

Agora a criação e preenchimento do vetor cronológico

```
let vd = new Array(36600).fill(0)
function montavet(){
   vd[0]=01011970
   i=0
   while (i<36600){
      vd[i+1]=amanha(vd[i])
   i++
   }
}
```

A manipulação das datas no aplicativo é

```
function execu(){
   dt1=document.getElementById('dt1').value;
   dt2=document.getElementById('dt2').value;
   dur=document.getElementById('dur').value;
   d1=dt1.split('/')
   d1=d1.map(Number)
   x1=(d1[0]*1000000)+(d1[1]*10000)+d1[2]
   i1=vd.indexOf(x1)
   if (dur==""){
      d2=dt2.split('/')
      d2=d2.map(Number)
      x2=(d2[0]*1000000)+(d2[1]*10000)+d2[2]
   i2=vd.indexOf(x2)
   console.log('i2='+i2)
   du=i2-i1
   document.getElementById('dur').value=du
      }
 else {
      i2=i1+parseInt(dur)
      df=Math.floor(vd[i2]/1000000)
      mf=Math.floor((vd[i2]-(df*1000000))/10000)
      af=vd[i2]%10000
   document.getElementById('dt2').value=
          df+'/'+mf+'/'+af
   }
}
```

☞ Para você fazer

Escreva o aplicativo aqui descrito. Mostre para o professor. Atente que haverá solicitações *ad hoc* durante a avaliação. Para testar seu aplicativo, use os exemplos dados na folha, ou quaisquer outros.

Avaliação:

Depois que programar o aplicativo calcule quantos dias há entre 28/ 7/2002 e 6/ 8/2008 .

Dada a data 3/ 8/1984 qual a data obtida depois de 1949 dias?

Responda aqui:

Dias de diferença	Data achada

404-76687 - /

Operações com datas

Mais um exercício para desenvolver as habilidades até agora trabalhadas, a saber: HTML, CSS e finalmente a programação Javascript. O aplicativo é para fazer cálculos envolvendo datas.

Mas para isso, vamos trabalhar alguns conceitos em JS, vejamos

Função de inicialização Se você precisar executar algo uma única vez ao carregar a página, um caminho possível é

- Definir `<body onload="inicializa()">` que informa que quando a página for carregada, deve ser executada a função `inicializa`.
- No script correspondente, defina a função `inicializa` fazendo o que deve ser feito.

Separar elementos de uma lista Supondo uma string com elementos separados por barras o comando `xx=lista.split('/')` vai criar uma lista onde cada elemento da string original será um elemento da lista

Converter lista de strings em lista de números Agora, o passo seguinte é converter cada elemento em um número. O comando é `yy=xx.map(Number)`. A função `map` aplica a função citada (neste caso é a `Number`) a cada elemento da lista.

Lista de saída No exercício desta folha você eventualmente precisa definir uma data de saída. A solução é

```
resp=' '
resp=dia+'/'+mes+"/"+ano
```

Note que o JS é suficientemente moderno para entender que valor é numérico mas como há strings na expressão (`"/"`) o sinal de adição não é adição e sim concatenação.

A encomenda

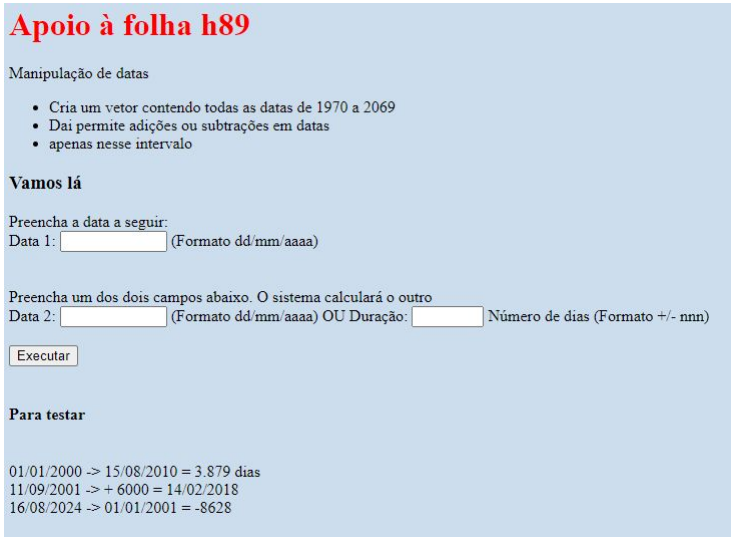
Note que a primeira coisa que o aplicativo pede é a digitação de uma data, em formato válido (`dd/mm/aaaa`) e compreendida no período válido (entre 1971 e 2069).

A seguir, o usuário pode escolher uma de duas opções

- Se optar por uma segunda data, o aplicativo vai calcular quantos dias separam as duas datas
- Se optar por uma quantidade (positiva ou negativa) de dias o sistema vai calcular a data obtida somando ou subtraindo a quantidade de dias informados em relação à primeira data informada.

A estratégia Devido à irregularidade do calendário, é muito difícil (e portanto sujeito a erros) a manipulação direta das variáveis dia, mês e ano em processos de adição e subtração de datas.

Analise a tela do seu aplicativo



Uma boa estratégia (certamente não a única) é criar um vetor de dias, onde o valor anterior corresponde à data de ontem, o atual é o dia de hoje e o próximo é amanha. Vamos chamar este vetor de vetor cronológico. Tendo construído este vetor (que é fixo e não precisa ser recalculado a cada momento), somar e subtrair datas se transforma no problema muito mais simples de somar e subtrair índices dentro desse vetor.

Guardar uma data em um inteiro Posto que uma data é um conjunto de 3 valores (dia, mês e ano, os 3 inteiros), para simplificar a construção do vetor cronológico, é conveniente combinar uma maneira de guardar os 3 valores em um único inteiro. Isto pode ser feito aplicando a seguinte expressão:

$$data = (dia \times 1000000) + (mes \times 10000) + ano$$

Para a construção do vetor cronológico é preciso estabelecer (combinar) com todos os que o usarão, qual o dia inicial do vetor (ou seja qual o valor de vetor[0]). Nada poderá ser feito com datas anteriores a esta. Nesta folha, vai-se usar como data inicial 1^o de janeiro de 1970, que vem a ser a data inicial de várias versões do sistema operacional Unix. Esta folha também vai realizar operações em 100 anos a partir dessa data, indo portanto até o ano de 2070. Diante disso, o tamanho do vetor cronológico é de mais ou menos 36600 elementos ($366 \times 100 = 36600$).

Próximo dia A função chave para construir o vetor cronológico é escrever uma função de nome amanhã, que receba uma data válida e devolva a data que corresponde a amanhã.

```
function amanha(hoje){
 dia=Math.floor(hoje/1000000)
 mes=Math.floor((hoje-(dia*1000000))/10000)
 ano=Math.floor(hoje-((dia*1000000)+(mes*10000)))
 if ((dia==31)&&(mes==12)){return 1010000+ano+1;}
 ud=0
 if ((dia==31)&&((mes==1)||(mes==3)||(mes==5)||
      (mes==7)||(mes==8)||(mes==10))){ud=1}
 if ((dia==30)&&((mes==4)||(mes==6)||(mes==9)||
      (mes==11))){ud=1}
 if ((ano%4==0)&&((ano%100!=0)||(ano%400==0))){
     bis=1}
 else {
     bis=0}
 if ((dia==28)&&(mes==2)&&(bis==0)){ud=1}
 if ((dia==29)&&(mes==2)&&(bis==1)){ud=1}
 if (ud==1){
    return 1000000+(10000*(mes+1))+ano
 }
 else {
    return (1000000*(dia+1))+(10000*mes)+ano
 }
}
```

Agora a criação e preenchimento do vetor cronológico

```
let vd = new Array(36600).fill(0)
function montavet(){
   vd[0]=01011970
   i=0
   while (i<36600){
      vd[i+1]=amanha(vd[i])
   i++
   }
}
```

A manipulação das datas no aplicativo é

```
function execu(){
   dt1=document.getElementById('dt1').value;
   dt2=document.getElementById('dt2').value;
   dur=document.getElementById('dur').value;
   d1=dt1.split('/')
   d1=d1.map(Number)
   x1=(d1[0]*1000000)+(d1[1]*10000)+d1[2]
   i1=vd.indexOf(x1)
   if (dur==""){
      d2=dt2.split('/')
      d2=d2.map(Number)
      x2=(d2[0]*1000000)+(d2[1]*10000)+d2[2]
   i2=vd.indexOf(x2)
   console.log('i2='+i2)
   du=i2-i1
   document.getElementById('dur').value=du
      }
 else {
      i2=i1+parseInt(dur)
      df=Math.floor(vd[i2]/1000000)
      mf=Math.floor((vd[i2]-(df*1000000))/10000)
      af=vd[i2]%10000
   document.getElementById('dt2').value=
           df+'/'+mf+'/'+af
   }
}
```

☞ Para você fazer

Escreva o aplicativo aqui descrito. Mostre para o professor. Atente que haverá solicitações *ad hoc* durante a avaliação. Para testar seu aplicativo, use os exemplos dados na folha, ou quaisquer outros.

Avaliação:

Depois que programar o aplicativo calcule quantos dias há entre 10/ 1/1984 e 17/ 1/1987 .

Dada a data 1/ 6/1978 qual a data obtida depois de 1444 dias?

Responda aqui:

Dias de diferença	Data achada

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 30/08/2024 - 11:20:57.0
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Operações com datas VIVXh89a V: 1.02
76113 NICOLE DA SILVA DE AZEVEDO
24FRO404 - 12 _____ / _____ / _____

Operações com datas

Mais um exercício para desenvolver as habilidades até agora trabalhadas, a saber: HTML, CSS e finalmente a programação Javascript. O aplicativo é para fazer cálculos envolvendo datas.

Mas para isso, vamos trabalhar alguns conceitos em JS, vejamos

Função de inicialização Se você precisar executar algo uma única vez ao carregar a página, um caminho possível é

- Definir `<body onload="inicializa()">` que informa que quando a página for carregada, deve ser executada a função `inicializa`.
- No script correspondente, defina a função `inicializa` fazendo o que deve ser feito.

Separar elementos de uma lista Supondo uma string com elementos separados por barras o comando `xx=lista.split('/')` vai criar uma lista onde cada elemento da string original será um elemento da lista

Converter lista de strings em lista de números Agora, o passo seguinte é converter cada elemento em um número. O comando é `yy=xx.map(Number)`. A função `map` aplica a função citada (neste caso é a `Number`) a cada elemento da lista.

Lista de saída No exercício desta folha você eventualmente precisa definir uma data de saída. A solução é

```
resp=' '
resp=dia+'/'+mes+"/"+ano
```

Note que o JS é suficientemente moderno para entender que valor é numérico mas como há strings na expressão (`"/"`) o sinal de adição não é adição e sim concatenação.

A encomenda

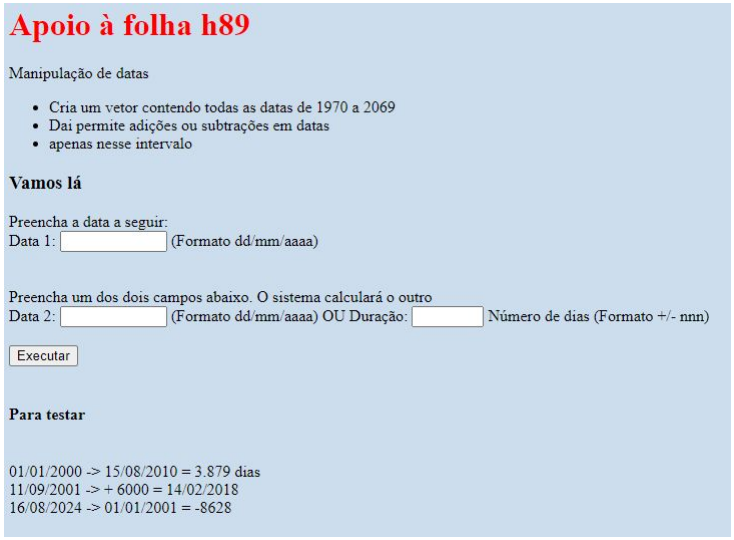
Note que a primeira coisa que o aplicativo pede é a digitação de uma data, em formato válido (`dd/mm/aaaa`) e compreendida no período válido (entre 1971 e 2069).

A seguir, o usuário pode escolher uma de duas opções

- Se optar por uma segunda data, o aplicativo vai calcular quantos dias separam as duas datas
- Se optar por uma quantidade (positiva ou negativa) de dias o sistema vai calcular a data obtida somando ou subtraindo a quantidade de dias informados em relação à primeira data informada.

A estratégia Devido à irregularidade do calendário, é muito difícil (e portanto sujeito a erros) a manipulação direta das variáveis dia, mês e ano em processos de adição e subtração de datas.

Analise a tela do seu aplicativo



Uma boa estratégia (certamente não a única) é criar um vetor de dias, onde o valor anterior corresponde à data de ontem, o atual é o dia de hoje e o próximo é amanha. Vamos chamar este vetor de vetor cronológico. Tendo construído este vetor (que é fixo e não precisa ser recalculado a cada momento), somar e subtrair datas se transforma no problema muito mais simples de somar e subtrair índices dentro desse vetor.

Guardar uma data em um inteiro Posto que uma data é um conjunto de 3 valores (dia, mês e ano, os 3 inteiros), para simplificar a construção do vetor cronológico, é conveniente combinar uma maneira de guardar os 3 valores em um único inteiro. Isto pode ser feito aplicando a seguinte expressão:

$$data = (dia \times 1000000) + (mes \times 10000) + ano$$

Para a construção do vetor cronológico é preciso estabelecer (combinar) com todos os que o usarão, qual o dia inicial do vetor (ou seja qual o valor de vetor[0]). Nada poderá ser feito com datas anteriores a esta. Nesta folha, vai-se usar como data inicial 1^o de janeiro de 1970, que vem a ser a data inicial de várias versões do sistema operacional Unix. Esta folha também vai realizar operações em 100 anos a partir dessa data, indo portanto até o ano de 2070. Diante disso, o tamanho do vetor cronológico é de mais ou menos 36600 elementos ($366 \times 100 = 36600$).

Próximo dia A função chave para construir o vetor cronológico é escrever uma função de nome amanhã, que receba uma data válida e devolva a data que corresponde a amanhã.

```
function amanha(hoje){
 dia=Math.floor(hoje/1000000)
 mes=Math.floor((hoje-(dia*1000000))/10000)
 ano=Math.floor(hoje-((dia*1000000)+(mes*10000)))
 if ((dia==31)&&(mes==12)){return 1010000+ano+1;}
 ud=0
 if ((dia==31)&&((mes==1)||(mes==3)||(mes==5)||
      (mes==7)||(mes==8)||(mes==10))){ud=1}
 if ((dia==30)&&((mes==4)||(mes==6)||(mes==9)||
      (mes==11))){ud=1}
 if ((ano%4==0)&&((ano%100!=0)||(ano%400==0))){
     bis=1}
 else {
     bis=0}
 if ((dia==28)&&(mes==2)&&(bis==0)){ud=1}
 if ((dia==29)&&(mes==2)&&(bis==1)){ud=1}
 if (ud==1){
    return 1000000+(10000*(mes+1))+ano
 }
 else {
    return (1000000*(dia+1))+(10000*mes)+ano
 }
}
```

Agora a criação e preenchimento do vetor cronológico

```
let vd = new Array(36600).fill(0)
function montavet(){
   vd[0]=01011970
   i=0
   while (i<36600){
      vd[i+1]=amanha(vd[i])
   i++
   }
}
```

A manipulação das datas no aplicativo é

```
function execu(){
   dt1=document.getElementById('dt1').value;
   dt2=document.getElementById('dt2').value;
   dur=document.getElementById('dur').value;
   d1=dt1.split('/')
   d1=d1.map(Number)
   x1=(d1[0]*1000000)+(d1[1]*10000)+d1[2]
   i1=vd.indexOf(x1)
   if (dur==""){
      d2=dt2.split('/')
      d2=d2.map(Number)
      x2=(d2[0]*1000000)+(d2[1]*10000)+d2[2]
   i2=vd.indexOf(x2)
   console.log('i2='+i2)
   du=i2-i1
   document.getElementById('dur').value=du
      }
 else {
      i2=i1+parseInt(dur)
      df=Math.floor(vd[i2]/1000000)
      mf=Math.floor((vd[i2]-(df*1000000))/10000)
      af=vd[i2]%10000
   document.getElementById('dt2').value=
           df+'/'+mf+'/'+af
   }
}
```

☞ Para você fazer

Escreva o aplicativo aqui descrito. Mostre para o professor. Atente que haverá solicitações *ad hoc* durante a avaliação. Para testar seu aplicativo, use os exemplos dados na folha, ou quaisquer outros.

Avaliação:

Depois que programar o aplicativo calcule quantos dias há entre 26/ 8/1982 e 19/ 5/1986 .

Dada a data 22/ 2/1988 qual a data obtida depois de 1299 dias?

Responda aqui:

Dias de diferença	Data achada

404-76113 - /

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 30/08/2024 - 11:20:57.0
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Operações com datas VIVXh89a V: 1.02
76120 PAULO JOSE FREIRE CORREA
24FRO404 - 13 _____ / _____ / _____

Operações com datas

Mais um exercício para desenvolver as habilidades até agora trabalhadas, a saber: HTML, CSS e finalmente a programação Javascript. O aplicativo é para fazer cálculos envolvendo datas.

Mas para isso, vamos trabalhar alguns conceitos em JS, vejamos

Função de inicialização Se você precisar executar algo uma única vez ao carregar a página, um caminho possível é

- Definir `<body onload="inicializa()">` que informa que quando a página for carregada, deve ser executada a função `inicializa`.
- No script correspondente, defina a função `inicializa` fazendo o que deve ser feito.

Separar elementos de uma lista Supondo uma string com elementos separados por barras o comando `xx=lista.split('/')` vai criar uma lista onde cada elemento da string original será um elemento da lista

Converter lista de strings em lista de números Agora, o passo seguinte é converter cada elemento em um número. O comando é `yy=xx.map(Number)`. A função `map` aplica a função citada (neste caso é a `Number`) a cada elemento da lista.

Lista de saída No exercício desta folha você eventualmente precisa definir uma data de saída. A solução é

```
resp=' '
resp=dia+'/'+mes+"/"+ano
```

Note que o JS é suficientemente moderno para entender que valor é numérico mas como há strings na expressão (`"/"`) o sinal de adição não é adição e sim concatenação.

A encomenda

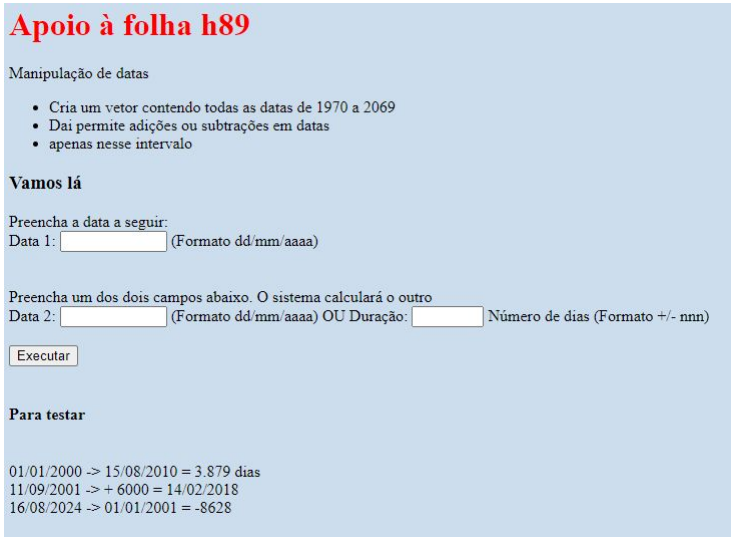
Note que a primeira coisa que o aplicativo pede é a digitação de uma data, em formato válido (`dd/mm/aaaa`) e compreendida no período válido (entre 1971 e 2069).

A seguir, o usuário pode escolher uma de duas opções

- Se optar por uma segunda data, o aplicativo vai calcular quantos dias separam as duas datas
- Se optar por uma quantidade (positiva ou negativa) de dias o sistema vai calcular a data obtida somando ou subtraindo a quantidade de dias informados em relação à primeira data informada.

A estratégia Devido à irregularidade do calendário, é muito difícil (e portanto sujeito a erros) a manipulação direta das variáveis dia, mês e ano em processos de adição e subtração de datas.

Analise a tela do seu aplicativo



Uma boa estratégia (certamente não a única) é criar um vetor de dias, onde o valor anterior corresponde à data de ontem, o atual é o dia de hoje e o próximo é amanha. Vamos chamar este vetor de vetor cronológico. Tendo construído este vetor (que é fixo e não precisa ser recalculado a cada momento), somar e subtrair datas se transforma no problema muito mais simples de somar e subtrair índices dentro desse vetor.

Guardar uma data em um inteiro Posto que uma data é um conjunto de 3 valores (dia, mês e ano, os 3 inteiros), para simplificar a construção do vetor cronológico, é conveniente combinar uma maneira de guardar os 3 valores em um único inteiro. Isto pode ser feito aplicando a seguinte expressão:

$$data = (dia \times 1000000) + (mes \times 10000) + ano$$

Para a construção do vetor cronológico é preciso estabelecer (combinar) com todos os que o usarão, qual o dia inicial do vetor (ou seja qual o valor de vetor[0]). Nada poderá ser feito com datas anteriores a esta. Nesta folha, vai-se usar como data inicial 1^o de janeiro de 1970, que vem a ser a data inicial de várias versões do sistema operacional Unix. Esta folha também vai realizar operações em 100 anos a partir dessa data, indo portanto até o ano de 2070. Diante disso, o tamanho do vetor cronológico é de mais ou menos 36600 elementos ($366 \times 100 = 36600$).

Próximo dia A função chave para construir o vetor cronológico é escrever uma função de nome amanhã, que receba uma data válida e devolva a data que corresponde a amanhã.

```
function amanha(hoje){
 dia=Math.floor(hoje/1000000)
 mes=Math.floor((hoje-(dia*1000000))/10000)
 ano=Math.floor(hoje-((dia*1000000)+(mes*10000)))
 if ((dia==31)&&(mes==12)){return 1010000+ano+1;}
 ud=0
 if ((dia==31)&&((mes==1)||(mes==3)||(mes==5)||
      (mes==7)||(mes==8)||(mes==10))){ud=1}
 if ((dia==30)&&((mes==4)||(mes==6)||(mes==9)||
      (mes==11))){ud=1}
 if ((ano%4==0)&&((ano%100!=0)||(ano%400==0))){
     bis=1}
 else {
     bis=0}
 if ((dia==28)&&(mes==2)&&(bis==0)){ud=1}
 if ((dia==29)&&(mes==2)&&(bis==1)){ud=1}
 if (ud==1){
    return 1000000+(10000*(mes+1))+ano
 }
 else {
    return (1000000*(dia+1))+(10000*mes)+ano
 }
}
```

Agora a criação e preenchimento do vetor cronológico

```
let vd = new Array(36600).fill(0)
function montavet(){
   vd[0]=01011970
   i=0
   while (i<36600){
      vd[i+1]=amanha(vd[i])
   i++
   }
}
```

A manipulação das datas no aplicativo é

```
function execu(){
   dt1=document.getElementById('dt1').value;
   dt2=document.getElementById('dt2').value;
   dur=document.getElementById('dur').value;
   d1=dt1.split('/')
   d1=d1.map(Number)
   x1=(d1[0]*1000000)+(d1[1]*10000)+d1[2]
   i1=vd.indexOf(x1)
   if (dur==""){
      d2=dt2.split('/')
      d2=d2.map(Number)
      x2=(d2[0]*1000000)+(d2[1]*10000)+d2[2]
   i2=vd.indexOf(x2)
   console.log('i2='+i2)
   du=i2-i1
   document.getElementById('dur').value=du
      }
 else {
      i2=i1+parseInt(dur)
      df=Math.floor(vd[i2]/1000000)
      mf=Math.floor((vd[i2]-(df*1000000))/10000)
      af=vd[i2]%10000
   document.getElementById('dt2').value=
          df+'/'+mf+'/'+af
   }
}
```

☞ Para você fazer

Escreva o aplicativo aqui descrito. Mostre para o professor. Atente que haverá solicitações *ad hoc* durante a avaliação. Para testar seu aplicativo, use os exemplos dados na folha, ou quaisquer outros.

Avaliação:

Depois que programar o aplicativo calcule quantos dias há entre 24/11/1982 e 22/12/1985 .

Dada a data 8/ 1/1999 qual a data obtida depois de 1574 dias?

Responda aqui:

Dias de diferença	Data achada

404-76120 - /

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 30/08/2024 - 11:20:57.0
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Operações com datas VIVXh89a V: 1.02
76137 PEDRO HENRIQUE PIEKARSKI DE 24FRO404 - 14 _____ / _____ / _____

Operações com datas

Mais um exercício para desenvolver as habilidades até agora trabalhadas, a saber: HTML, CSS e finalmente a programação Javascript. O aplicativo é para fazer cálculos envolvendo datas.

Mas para isso, vamos trabalhar alguns conceitos em JS, vejamos

Função de inicialização Se você precisar executar algo uma única vez ao carregar a página, um caminho possível é

- Definir `<body onload="inicializa()">` que informa que quando a página for carregada, deve ser executada a função `inicializa`.
- No script correspondente, defina a função `inicializa` fazendo o que deve ser feito.

Separar elementos de uma lista Supondo uma string com elementos separados por barras o comando `xx=lista.split('/')` vai criar uma lista onde cada elemento da string original será um elemento da lista

Converter lista de strings em lista de números Agora, o passo seguinte é converter cada elemento em um número. O comando é `yy=xx.map(Number)`. A função `map` aplica a função citada (neste caso é a `Number`) a cada elemento da lista.

Lista de saída No exercício desta folha você eventualmente precisa definir uma data de saída. A solução é

```
resp=' '
resp=dia+'/'+mes+"/"+ano
```

Note que o JS é suficientemente moderno para entender que valor é numérico mas como há strings na expressão (`"/"`) o sinal de adição não é adição e sim concatenação.

A encomenda

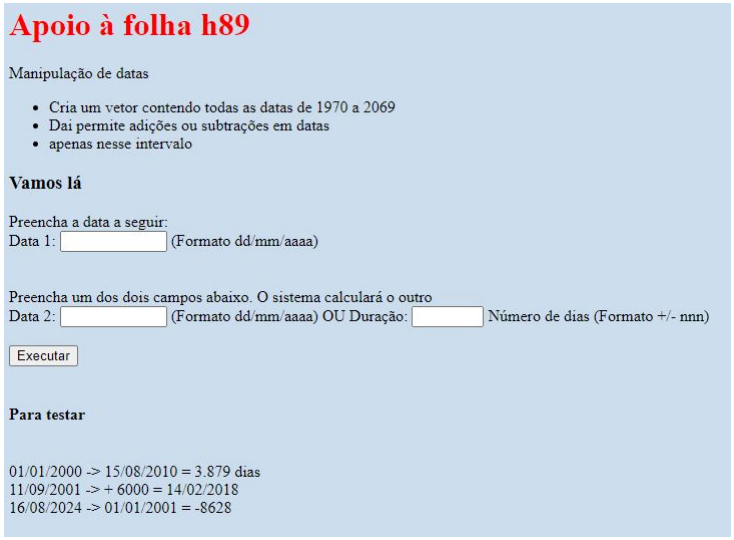
Note que a primeira coisa que o aplicativo pede é a digitação de uma data, em formato válido (`dd/mm/aaaa`) e compreendida no período válido (entre 1971 e 2069).

A seguir, o usuário pode escolher uma de duas opções

- Se optar por uma segunda data, o aplicativo vai calcular quantos dias separam as duas datas
- Se optar por uma quantidade (positiva ou negativa) de dias o sistema vai calcular a data obtida somando ou subtraindo a quantidade de dias informados em relação à primeira data informada.

A estratégia Devido à irregularidade do calendário, é muito difícil (e portanto sujeito a erros) a manipulação direta das variáveis dia, mês e ano em processos de adição e subtração de datas.

Analise a tela do seu aplicativo



Uma boa estratégia (certamente não a única) é criar um vetor de dias, onde o valor anterior corresponde à data de ontem, o atual é o dia de hoje e o próximo é amanha. Vamos chamar este vetor de vetor cronológico. Tendo construído este vetor (que é fixo e não precisa ser recalculado a cada momento), somar e subtrair datas se transforma no problema muito mais simples de somar e subtrair índices dentro desse vetor.

Guardar uma data em um inteiro Posto que uma data é um conjunto de 3 valores (dia, mês e ano, os 3 inteiros), para simplificar a construção do vetor cronológico, é conveniente combinar uma maneira de guardar os 3 valores em um único inteiro. Isto pode ser feito aplicando a seguinte expressão:

$$data = (dia \times 1000000) + (mes \times 10000) + ano$$

Para a construção do vetor cronológico é preciso estabelecer (combinar) com todos os que o usarão, qual o dia inicial do vetor (ou seja qual o valor de vetor[0]). Nada poderá ser feito com datas anteriores a esta. Nesta folha, vai-se usar como data inicial 1^o de janeiro de 1970, que vem a ser a data inicial de várias versões do sistema operacional Unix. Esta folha também vai realizar operações em 100 anos a partir dessa data, indo portanto até o ano de 2070. Diante disso, o tamanho do vetor cronológico é de mais ou menos 36600 elementos ($366 \times 100 = 36600$).

Próximo dia A função chave para construir o vetor cronológico é escrever uma função de nome amanhã, que receba uma data válida e devolva a data que corresponde a amanhã.

```
function amanha(hoje){
 dia=Math.floor(hoje/1000000)
 mes=Math.floor((hoje-(dia*1000000))/10000)
 ano=Math.floor(hoje-((dia*1000000)+(mes*10000)))
 if ((dia==31)&&(mes==12)){return 1010000+ano+1;}
 ud=0
 if ((dia==31)&&((mes==1)||(mes==3)||(mes==5)||
      (mes==7)||(mes==8)||(mes==10))){ud=1}
 if ((dia==30)&&((mes==4)||(mes==6)||(mes==9)||
      (mes==11))){ud=1}
 if ((ano%4==0)&&((ano%100!=0)||(ano%400==0))){
     bis=1}
 else {
     bis=0}
 if ((dia==28)&&(mes==2)&&(bis==0)){ud=1}
 if ((dia==29)&&(mes==2)&&(bis==1)){ud=1}
 if (ud==1){
    return 1000000+(10000*(mes+1))+ano
 }
 else {
    return (1000000*(dia+1))+(10000*mes)+ano
 }
}
```

Agora a criação e preenchimento do vetor cronológico

```
let vd = new Array(36600).fill(0)
function montavet(){
   vd[0]=01011970
   i=0
   while (i<36600){
      vd[i+1]=amanha(vd[i])
   i++
   }
}
```

A manipulação das datas no aplicativo é

```
function execu(){
   dt1=document.getElementById('dt1').value;
   dt2=document.getElementById('dt2').value;
   dur=document.getElementById('dur').value;
   d1=dt1.split('/')
   d1=d1.map(Number)
   x1=(d1[0]*1000000)+(d1[1]*10000)+d1[2]
   i1=vd.indexOf(x1)
   if (dur==""){
      d2=dt2.split('/')
      d2=d2.map(Number)
      x2=(d2[0]*1000000)+(d2[1]*10000)+d2[2]
   i2=vd.indexOf(x2)
   console.log('i2='+i2)
   du=i2-i1
   document.getElementById('dur').value=du
      }
 else {
      i2=i1+parseInt(dur)
      df=Math.floor(vd[i2]/1000000)
      mf=Math.floor((vd[i2]-(df*1000000))/10000)
      af=vd[i2]%10000
   document.getElementById('dt2').value=
          df+'/'+mf+'/'+af
   }
}
```

☞ Para você fazer

Escreva o aplicativo aqui descrito. Mostre para o professor. Atente que haverá solicitações *ad hoc* durante a avaliação. Para testar seu aplicativo, use os exemplos dados na folha, ou quaisquer outros.

Avaliação:

Depois que programar o aplicativo calcule quantos dias há entre 14/ 2/2002 e 15/ 9/2007 .

Dada a data 30/12/1981 qual a data obtida depois de 1189 dias?

Responda aqui:

Dias de diferença	Data achada

404-76137 - /

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 30/08/2024 - 11:20:57.0
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Operações com datas VIVXh89a V: 1.02
76144 RAUL KNAPKI CUNHA
24FRO404 - 15 _____ / _____ / _____

Operações com datas

Mais um exercício para desenvolver as habilidades até agora trabalhadas, a saber: HTML, CSS e finalmente a programação Javascript. O aplicativo é para fazer cálculos envolvendo datas.

Mas para isso, vamos trabalhar alguns conceitos em JS, vejamos

Função de inicialização Se você precisar executar algo uma única vez ao carregar a página, um caminho possível é

- Definir `<body onload="inicializa()">` que informa que quando a página for carregada, deve ser executada a função `inicializa`.
- No script correspondente, defina a função `inicializa` fazendo o que deve ser feito.

Separar elementos de uma lista Supondo uma string com elementos separados por barras o comando `xx=lista.split('/')` vai criar uma lista onde cada elemento da string original será um elemento da lista

Converter lista de strings em lista de números Agora, o passo seguinte é converter cada elemento em um número. O comando é `yy=xx.map(Number)`. A função `map` aplica a função citada (neste caso é a `Number`) a cada elemento da lista.

Lista de saída No exercício desta folha você eventualmente precisa definir uma data de saída. A solução é

```
resp=' '
resp=dia+'/'+mes+"/"+ano
```

Note que o JS é suficientemente moderno para entender que valor é numérico mas como há strings na expressão (`"/"`) o sinal de adição não é adição e sim concatenação.

A encomenda

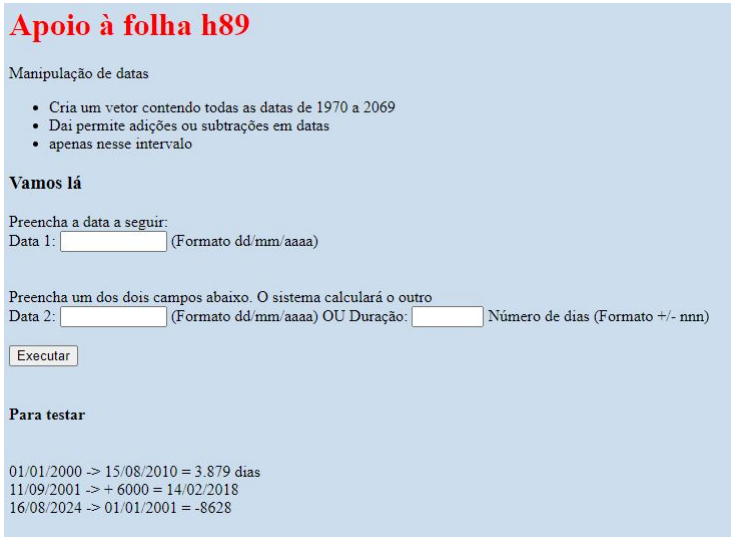
Note que a primeira coisa que o aplicativo pede é a digitação de uma data, em formato válido (`dd/mm/aaaa`) e compreendida no período válido (entre 1971 e 2069).

A seguir, o usuário pode escolher uma de duas opções

- Se optar por uma segunda data, o aplicativo vai calcular quantos dias separam as duas datas
- Se optar por uma quantidade (positiva ou negativa) de dias o sistema vai calcular a data obtida somando ou subtraindo a quantidade de dias informados em relação à primeira data informada.

A estratégia Devido à irregularidade do calendário, é muito difícil (e portanto sujeito a erros) a manipulação direta das variáveis dia, mês e ano em processos de adição e subtração de datas.

Analise a tela do seu aplicativo



Uma boa estratégia (certamente não a única) é criar um vetor de dias, onde o valor anterior corresponde à data de ontem, o atual é o dia de hoje e o próximo é amanha. Vamos chamar este vetor de vetor cronológico. Tendo construído este vetor (que é fixo e não precisa ser recalculado a cada momento), somar e subtrair datas se transforma no problema muito mais simples de somar e subtrair índices dentro desse vetor.

Guardar uma data em um inteiro Posto que uma data é um conjunto de 3 valores (dia, mês e ano, os 3 inteiros), para simplificar a construção do vetor cronológico, é conveniente combinar uma maneira de guardar os 3 valores em um único inteiro. Isto pode ser feito aplicando a seguinte expressão:

$$data = (dia \times 1000000) + (mes \times 10000) + ano$$

Para a construção do vetor cronológico é preciso estabelecer (combinar) com todos os que o usarão, qual o dia inicial do vetor (ou seja qual o valor de vetor[0]). Nada poderá ser feito com datas anteriores a esta. Nesta folha, vai-se usar como data inicial 1^o de janeiro de 1970, que vem a ser a data inicial de várias versões do sistema operacional Unix. Esta folha também vai realizar operações em 100 anos a partir dessa data, indo portanto até o ano de 2070. Diante disso, o tamanho do vetor cronológico é de mais ou menos 36600 elementos ($366 \times 100 = 36600$).

Próximo dia A função chave para construir o vetor cronológico é escrever uma função de nome amanhã, que receba uma data válida e devolva a data que corresponde a amanhã.

```
function amanha(hoje){
 dia=Math.floor(hoje/1000000)
 mes=Math.floor((hoje-(dia*1000000))/10000)
 ano=Math.floor(hoje-((dia*1000000)+(mes*10000)))
 if ((dia==31)&&(mes==12)){return 1010000+ano+1;}
 ud=0
 if ((dia==31)&&((mes==1)||(mes==3)||(mes==5)||
      (mes==7)||(mes==8)||(mes==10))){ud=1}
 if ((dia==30)&&((mes==4)||(mes==6)||(mes==9)||
      (mes==11))){ud=1}
 if ((ano%4==0)&&((ano%100!=0)||(ano%400==0))){
     bis=1}
 else {
     bis=0}
 if ((dia==28)&&(mes==2)&&(bis==0)){ud=1}
 if ((dia==29)&&(mes==2)&&(bis==1)){ud=1}
 if (ud==1){
    return 1000000+(10000*(mes+1))+ano
 }
 else {
    return (1000000*(dia+1))+(10000*mes)+ano
 }
}
```

Agora a criação e preenchimento do vetor cronológico

```
let vd = new Array(36600).fill(0)
function montavet(){
   vd[0]=01011970
   i=0
   while (i<36600){
      vd[i+1]=amanha(vd[i])
   i++
   }
}
```

A manipulação das datas no aplicativo é

```
function execu(){
   dt1=document.getElementById('dt1').value;
   dt2=document.getElementById('dt2').value;
   dur=document.getElementById('dur').value;
   d1=dt1.split('/')
   d1=d1.map(Number)
   x1=(d1[0]*1000000)+(d1[1]*10000)+d1[2]
   i1=vd.indexOf(x1)
   if (dur==""){
      d2=dt2.split('/')
      d2=d2.map(Number)
      x2=(d2[0]*1000000)+(d2[1]*10000)+d2[2]
   i2=vd.indexOf(x2)
   console.log('i2='+i2)
   du=i2-i1
   document.getElementById('dur').value=du
      }
 else {
      i2=i1+parseInt(dur)
      df=Math.floor(vd[i2]/1000000)
      mf=Math.floor((vd[i2]-(df*1000000))/10000)
      af=vd[i2]%10000
   document.getElementById('dt2').value=
           df+'/'+mf+'/'+af
   }
}
```

☞ Para você fazer

Escreva o aplicativo aqui descrito. Mostre para o professor. Atente que haverá solicitações *ad hoc* durante a avaliação. Para testar seu aplicativo, use os exemplos dados na folha, ou quaisquer outros.

Avaliação:

Depois que programar o aplicativo calcule quantos dias há entre 24/ 9/1997 e 4/ 7/2001 .

Dada a data 25/ 3/1989 qual a data obtida depois de 1584 dias?

Responda aqui:

Dias de diferença	Data achada

404-76144 - /

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 30/08/2024 - 11:20:57.0
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Operações com datas VIVXh89a V: 1.02
76168 VICTOR BRONDANI DOS SANTOS
24FRO404 - 16 _____ / _____ / _____

Operações com datas

Mais um exercício para desenvolver as habilidades até agora trabalhadas, a saber: HTML, CSS e finalmente a programação Javascript. O aplicativo é para fazer cálculos envolvendo datas.

Mas para isso, vamos trabalhar alguns conceitos em JS, vejamos

Função de inicialização Se você precisar executar algo uma única vez ao carregar a página, um caminho possível é

- Definir `<body onload="inicializa()">` que informa que quando a página for carregada, deve ser executada a função `inicializa`.
- No script correspondente, defina a função `inicializa` fazendo o que deve ser feito.

Separar elementos de uma lista Supondo uma string com elementos separados por barras o comando `xx=lista.split('/')` vai criar uma lista onde cada elemento da string original será um elemento da lista

Converter lista de strings em lista de números Agora, o passo seguinte é converter cada elemento em um número. O comando é `yy=xx.map(Number)`. A função `map` aplica a função citada (neste caso é a `Number`) a cada elemento da lista.

Lista de saída No exercício desta folha você eventualmente precisa definir uma data de saída. A solução é

```
resp=' '
resp=dia+'/'+mes+"/"+ano
```

Note que o JS é suficientemente moderno para entender que valor é numérico mas como há strings na expressão (`"/"`) o sinal de adição não é adição e sim concatenação.

A encomenda

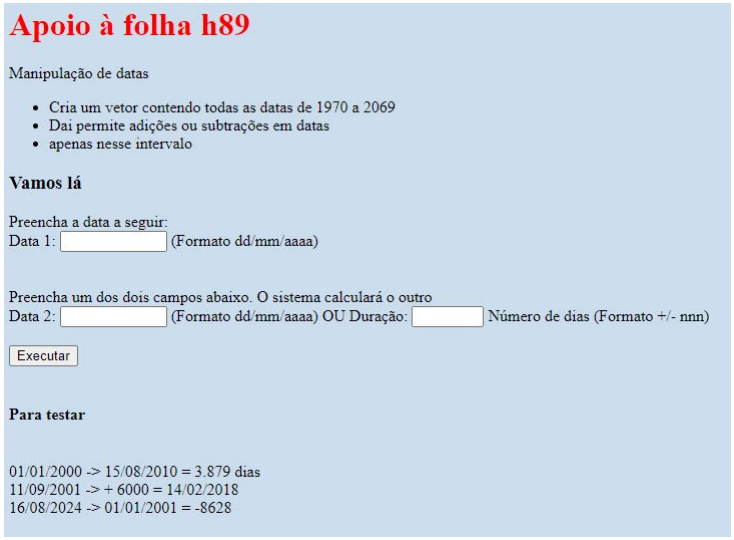
Note que a primeira coisa que o aplicativo pede é a digitação de uma data, em formato válido (`dd/mm/aaaa`) e compreendida no período válido (entre 1971 e 2069).

A seguir, o usuário pode escolher uma de duas opções

- Se optar por uma segunda data, o aplicativo vai calcular quantos dias separam as duas datas
- Se optar por uma quantidade (positiva ou negativa) de dias o sistema vai calcular a data obtida somando ou subtraindo a quantidade de dias informados em relação à primeira data informada.

A estratégia Devido à irregularidade do calendário, é muito difícil (e portanto sujeito a erros) a manipulação direta das variáveis dia, mês e ano em processos de adição e subtração de datas.

Analise a tela do seu aplicativo



Uma boa estratégia (certamente não a única) é criar um vetor de dias, onde o valor anterior corresponde à data de ontem, o atual é o dia de hoje e o próximo é amanha. Vamos chamar este vetor de vetor cronológico. Tendo construído este vetor (que é fixo e não precisa ser recalculado a cada momento), somar e subtrair datas se transforma no problema muito mais simples de somar e subtrair índices dentro desse vetor.

Guardar uma data em um inteiro Posto que uma data é um conjunto de 3 valores (dia, mês e ano, os 3 inteiros), para simplificar a construção do vetor cronológico, é conveniente combinar uma maneira de guardar os 3 valores em um único inteiro. Isto pode ser feito aplicando a seguinte expressão:

$$data = (dia \times 1000000) + (mes \times 10000) + ano$$

Para a construção do vetor cronológico é preciso estabelecer (combinar) com todos os que o usarão, qual o dia inicial do vetor (ou seja qual o valor de vetor[0]). Nada poderá ser feito com datas anteriores a esta. Nesta folha, vai-se usar como data inicial 1^o de janeiro de 1970, que vem a ser a data inicial de várias versões do sistema operacional Unix. Esta folha também vai realizar operações em 100 anos a partir dessa data, indo portanto até o ano de 2070. Diante disso, o tamanho do vetor cronológico é de mais ou menos 36600 elementos ($366 \times 100 = 36600$).

Próximo dia A função chave para construir o vetor cronológico é escrever uma função de nome amanhã, que receba uma data válida e devolva a data que corresponde a amanhã.

```
function amanha(hoje){
 dia=Math.floor(hoje/1000000)
 mes=Math.floor((hoje-(dia*1000000))/10000)
 ano=Math.floor(hoje-((dia*1000000)+(mes*10000)))
 if ((dia==31)&&(mes==12)){return 1010000+ano+1;}
 ud=0
 if ((dia==31)&&((mes==1)||(mes==3)||(mes==5)||
      (mes==7)||(mes==8)||(mes==10))){ud=1}
 if ((dia==30)&&((mes==4)||(mes==6)||(mes==9)||
      (mes==11))){ud=1}
 if ((ano%4==0)&&((ano%100!=0)||(ano%400==0))){
     bis=1}
 else {
     bis=0}
 if ((dia==28)&&(mes==2)&&(bis==0)){ud=1}
 if ((dia==29)&&(mes==2)&&(bis==1)){ud=1}
 if (ud==1){
    return 1000000+(10000*(mes+1))+ano
 }
 else {
    return (1000000*(dia+1))+(10000*mes)+ano
 }
}
```

Agora a criação e preenchimento do vetor cronológico

```
let vd = new Array(36600).fill(0)
function montavet(){
   vd[0]=01011970
   i=0
   while (i<36600){
      vd[i+1]=amanha(vd[i])
   i++
   }
}
```

A manipulação das datas no aplicativo é

```
function execu(){
   dt1=document.getElementById('dt1').value;
   dt2=document.getElementById('dt2').value;
   dur=document.getElementById('dur').value;
   d1=dt1.split('/')
   d1=d1.map(Number)
   x1=(d1[0]*1000000)+(d1[1]*10000)+d1[2]
   i1=vd.indexOf(x1)
   if (dur==""){
      d2=dt2.split('/')
      d2=d2.map(Number)
      x2=(d2[0]*1000000)+(d2[1]*10000)+d2[2]
   i2=vd.indexOf(x2)
   console.log('i2='+i2)
   du=i2-i1
   document.getElementById('dur').value=du
      }
 else {
      i2=i1+parseInt(dur)
      df=Math.floor(vd[i2]/1000000)
      mf=Math.floor((vd[i2]-(df*1000000))/10000)
      af=vd[i2]%10000
   document.getElementById('dt2').value=
           df+'/'+mf+'/'+af
   }
}
```

☞ Para você fazer

Escreva o aplicativo aqui descrito. Mostre para o professor. Atente que haverá solicitações *ad hoc* durante a avaliação. Para testar seu aplicativo, use os exemplos dados na folha, ou quaisquer outros.

Avaliação:

Depois que programar o aplicativo calcule quantos dias há entre 9/ 5/1989 e 8/11/1996 .

Dada a data 27/ 7/1977 qual a data obtida depois de 1993 dias?

Responda aqui:

Dias de diferença	Data achada

404-76168 - /

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 30/08/2024 - 11:20:57.0
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Operações com datas VIVXh89a V: 1.02
76175 VICTOR HUGO DOS SANTOS DE CA
24FRO404 - 17 _____ / _____ / _____

Operações com datas

Mais um exercício para desenvolver as habilidades até agora trabalhadas, a saber: HTML, CSS e finalmente a programação Javascript. O aplicativo é para fazer cálculos envolvendo datas.

Mas para isso, vamos trabalhar alguns conceitos em JS, vejamos

Função de inicialização Se você precisar executar algo uma única vez ao carregar a página, um caminho possível é

- Definir `<body onload="inicializa()">` que informa que quando a página for carregada, deve ser executada a função `inicializa`.
- No script correspondente, defina a função `inicializa` fazendo o que deve ser feito.

Separar elementos de uma lista Supondo uma string com elementos separados por barras o comando `xx=lista.split('/')` vai criar uma lista onde cada elemento da string original será um elemento da lista

Converter lista de strings em lista de números Agora, o passo seguinte é converter cada elemento em um número. O comando é `yy=xx.map(Number)`. A função `map` aplica a função citada (neste caso é a `Number`) a cada elemento da lista.

Lista de saída No exercício desta folha você eventualmente precisa definir uma data de saída. A solução é

```
resp=' '
resp=dia+'/'+mes+"/"+ano
```

Note que o JS é suficientemente moderno para entender que valor é numérico mas como há strings na expressão ("/") o sinal de adição não é adição e sim concatenação.

A encomenda

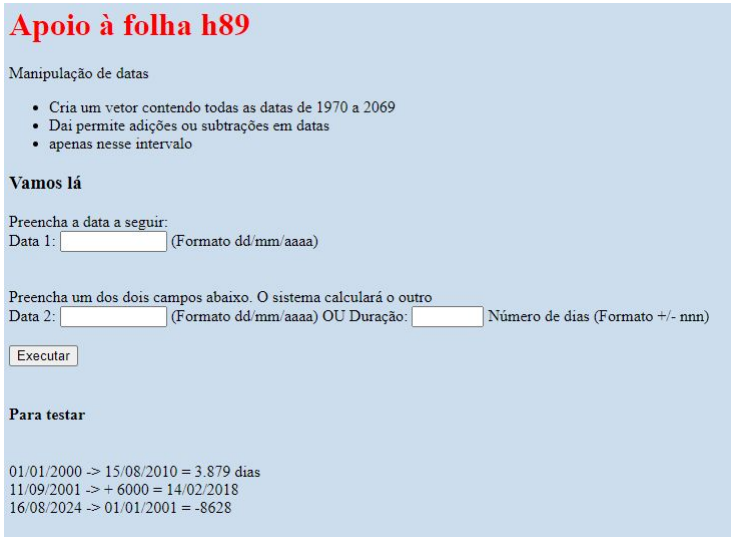
Note que a primeira coisa que o aplicativo pede é a digitação de uma data, em formato válido (`dd/mm/aaaa`) e compreendida no período válido (entre 1971 e 2069).

A seguir, o usuário pode escolher uma de duas opções

- Se optar por uma segunda data, o aplicativo vai calcular quantos dias separam as duas datas
- Se optar por uma quantidade (positiva ou negativa) de dias o sistema vai calcular a data obtida somando ou subtraindo a quantidade de dias informados em relação à primeira data informada.

A estratégia Devido à irregularidade do calendário, é muito difícil (e portanto sujeito a erros) a manipulação direta das variáveis dia, mês e ano em processos de adição e subtração de datas.

Analise a tela do seu aplicativo



Uma boa estratégia (certamente não a única) é criar um vetor de dias, onde o valor anterior corresponde à data de ontem, o atual é o dia de hoje e o próximo é amanha. Vamos chamar este vetor de vetor cronológico. Tendo construído este vetor (que é fixo e não precisa ser recalculado a cada momento), somar e subtrair datas se transforma no problema muito mais simples de somar e subtrair índices dentro desse vetor.

Guardar uma data em um inteiro Posto que uma data é um conjunto de 3 valores (dia, mês e ano, os 3 inteiros), para simplificar a construção do vetor cronológico, é conveniente combinar uma maneira de guardar os 3 valores em um único inteiro. Isto pode ser feito aplicando a seguinte expressão:

$$data = (dia \times 1000000) + (mes \times 10000) + ano$$

Para a construção do vetor cronológico é preciso estabelecer (combinar) com todos os que o usarão, qual o dia inicial do vetor (ou seja qual o valor de vetor[0]). Nada poderá ser feito com datas anteriores a esta. Nesta folha, vai-se usar como data inicial 1^o de janeiro de 1970, que vem a ser a data inicial de várias versões do sistema operacional Unix. Esta folha também vai realizar operações em 100 anos a partir dessa data, indo portanto até o ano de 2070. Diante disso, o tamanho do vetor cronológico é de mais ou menos 36600 elementos ($366 \times 100 = 36600$).

Próximo dia A função chave para construir o vetor cronológico é escrever uma função de nome amanhã, que receba uma data válida e devolva a data que corresponde a amanhã.

```
function amanha(hoje){
 dia=Math.floor(hoje/1000000)
 mes=Math.floor((hoje-(dia*1000000))/10000)
 ano=Math.floor(hoje-((dia*1000000)+(mes*10000)))
 if ((dia==31)&&(mes==12)){return 1010000+ano+1;}
 ud=0
 if ((dia==31)&&((mes==1)||(mes==3)||(mes==5)||
      (mes==7)||(mes==8)||(mes==10))){ud=1}
 if ((dia==30)&&((mes==4)||(mes==6)||(mes==9)||
      (mes==11))){ud=1}
 if ((ano%4==0)&&((ano%100!=0)||(ano%400==0))){
     bis=1}
 else {
     bis=0}
 if ((dia==28)&&(mes==2)&&(bis==0)){ud=1}
 if ((dia==29)&&(mes==2)&&(bis==1)){ud=1}
 if (ud==1){
    return 1000000+(10000*(mes+1))+ano
 }
 else {
    return (1000000*(dia+1))+(10000*mes)+ano
 }
}
```

Agora a criação e preenchimento do vetor cronológico

```
let vd = new Array(36600).fill(0)
function montavet(){
   vd[0]=01011970
   i=0
   while (i<36600){
      vd[i+1]=amanha(vd[i])
   i++
   }
}
```

A manipulação das datas no aplicativo é

```
function execu(){
   dt1=document.getElementById('dt1').value;
   dt2=document.getElementById('dt2').value;
   dur=document.getElementById('dur').value;
   d1=dt1.split('/')
   d1=d1.map(Number)
   x1=(d1[0]*1000000)+(d1[1]*10000)+d1[2]
   i1=vd.indexOf(x1)
   if (dur==""){
      d2=dt2.split('/')
      d2=d2.map(Number)
      x2=(d2[0]*1000000)+(d2[1]*10000)+d2[2]
   i2=vd.indexOf(x2)
   console.log('i2='+i2)
   du=i2-i1
   document.getElementById('dur').value=du
      }
 else {
      i2=i1+parseInt(dur)
      df=Math.floor(vd[i2]/1000000)
      mf=Math.floor((vd[i2]-(df*1000000))/10000)
      af=vd[i2]%10000
   document.getElementById('dt2').value=
          df+'/'+mf+'/'+af
   }
}
```

☞ Para você fazer

Escreva o aplicativo aqui descrito. Mostre para o professor. Atente que haverá solicitações *ad hoc* durante a avaliação. Para testar seu aplicativo, use os exemplos dados na folha, ou quaisquer outros.

Avaliação:

Depois que programar o aplicativo calcule quantos dias há entre 13/ 2/1978 e 7/ 8/1983 .

Dada a data 5/ 6/1999 qual a data obtida depois de 1476 dias?

Responda aqui:

Dias de diferença	Data achada

404-76175 - /

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 30/08/2024 - 11:20:57.0
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Operações com datas VIVXh89a V: 1.02
76182 VITOR EDUARDO DE GOES FONTES
24FRO404 - 18 _____ / _____ / _____

Operações com datas

Mais um exercício para desenvolver as habilidades até agora trabalhadas, a saber: HTML, CSS e finalmente a programação Javascript. O aplicativo é para fazer cálculos envolvendo datas.

Mas para isso, vamos trabalhar alguns conceitos em JS, vejamos

Função de inicialização Se você precisar executar algo uma única vez ao carregar a página, um caminho possível é

- Definir `<body onload="inicializa()">` que informa que quando a página for carregada, deve ser executada a função `inicializa`.
- No script correspondente, defina a função `inicializa` fazendo o que deve ser feito.

Separar elementos de uma lista Supondo uma string com elementos separados por barras o comando `xx=lista.split('/')` vai criar uma lista onde cada elemento da string original será um elemento da lista

Converter lista de strings em lista de números Agora, o passo seguinte é converter cada elemento em um número. O comando é `yy=xx.map(Number)`. A função `map` aplica a função citada (neste caso é a `Number`) a cada elemento da lista.

Lista de saída No exercício desta folha você eventualmente precisa definir uma data de saída. A solução é

```
resp=' '
resp=dia+'/'+mes+"/"+ano
```

Note que o JS é suficientemente moderno para entender que valor é numérico mas como há strings na expressão (`"/"`) o sinal de adição não é adição e sim concatenação.

A encomenda

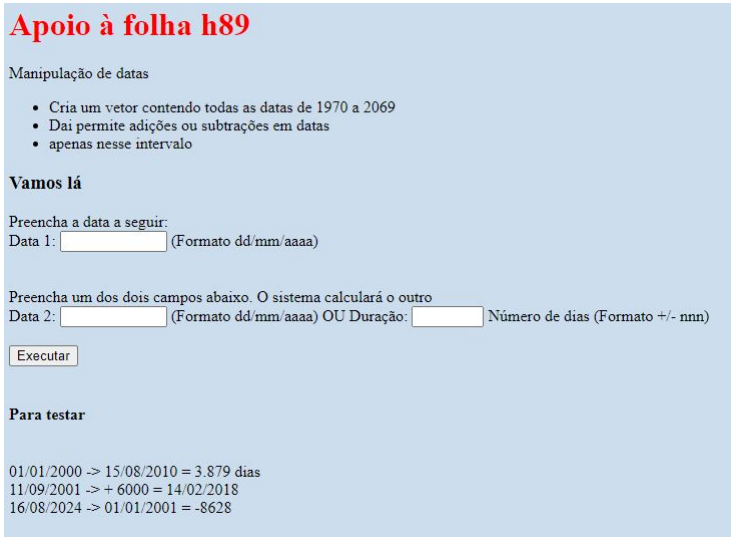
Note que a primeira coisa que o aplicativo pede é a digitação de uma data, em formato válido (`dd/mm/aaaa`) e compreendida no período válido (entre 1971 e 2069).

A seguir, o usuário pode escolher uma de duas opções

- Se optar por uma segunda data, o aplicativo vai calcular quantos dias separam as duas datas
- Se optar por uma quantidade (positiva ou negativa) de dias o sistema vai calcular a data obtida somando ou subtraindo a quantidade de dias informados em relação à primeira data informada.

A estratégia Devido à irregularidade do calendário, é muito difícil (e portanto sujeito a erros) a manipulação direta das variáveis dia, mês e ano em processos de adição e subtração de datas.

Analise a tela do seu aplicativo



Uma boa estratégia (certamente não a única) é criar um vetor de dias, onde o valor anterior corresponde à data de ontem, o atual é o dia de hoje e o próximo é amanha. Vamos chamar este vetor de vetor cronológico. Tendo construído este vetor (que é fixo e não precisa ser recalculado a cada momento), somar e subtrair datas se transforma no problema muito mais simples de somar e subtrair índices dentro desse vetor.

Guardar uma data em um inteiro Posto que uma data é um conjunto de 3 valores (dia, mês e ano, os 3 inteiros), para simplificar a construção do vetor cronológico, é conveniente combinar uma maneira de guardar os 3 valores em um único inteiro. Isto pode ser feito aplicando a seguinte expressão:

$$data = (dia \times 1000000) + (mes \times 10000) + ano$$

Para a construção do vetor cronológico é preciso estabelecer (combinar) com todos os que o usarão, qual o dia inicial do vetor (ou seja qual o valor de vetor[0]). Nada poderá ser feito com datas anteriores a esta. Nesta folha, vai-se usar como data inicial 1^o de janeiro de 1970, que vem a ser a data inicial de várias versões do sistema operacional Unix. Esta folha também vai realizar operações em 100 anos a partir dessa data, indo portanto até o ano de 2070. Diante disso, o tamanho do vetor cronológico é de mais ou menos 36600 elementos ($366 \times 100 = 36600$).

Próximo dia A função chave para construir o vetor cronológico é escrever uma função de nome amanhã, que receba uma data válida e devolva a data que corresponde a amanhã.

```
function amanha(hoje){
 dia=Math.floor(hoje/1000000)
 mes=Math.floor((hoje-(dia*1000000))/10000)
 ano=Math.floor(hoje-((dia*1000000)+(mes*10000)))
 if ((dia==31)&&(mes==12)){return 1010000+ano+1;}
 ud=0
 if ((dia==31)&&((mes==1)||(mes==3)||(mes==5)||
      (mes==7)||(mes==8)||(mes==10))){ud=1}
 if ((dia==30)&&((mes==4)||(mes==6)||(mes==9)||
      (mes==11))){ud=1}
 if ((ano%4==0)&&((ano%100!=0)||(ano%400==0))){
     bis=1}
 else {
     bis=0}
 if ((dia==28)&&(mes==2)&&(bis==0)){ud=1}
 if ((dia==29)&&(mes==2)&&(bis==1)){ud=1}
 if (ud==1){
    return 1000000+(10000*(mes+1))+ano
 }
 else {
    return (1000000*(dia+1))+(10000*mes)+ano
 }
}
```

Agora a criação e preenchimento do vetor cronológico

```
let vd = new Array(36600).fill(0)
function montavet(){
   vd[0]=01011970
   i=0
   while (i<36600){
      vd[i+1]=amanha(vd[i])
   i++
   }
}
```

A manipulação das datas no aplicativo é

```
function execu(){
   dt1=document.getElementById('dt1').value;
   dt2=document.getElementById('dt2').value;
   dur=document.getElementById('dur').value;
   d1=dt1.split('/')
   d1=d1.map(Number)
   x1=(d1[0]*1000000)+(d1[1]*10000)+d1[2]
   i1=vd.indexOf(x1)
   if (dur==""){
      d2=dt2.split('/')
      d2=d2.map(Number)
      x2=(d2[0]*1000000)+(d2[1]*10000)+d2[2]
   i2=vd.indexOf(x2)
   console.log('i2='+i2)
   du=i2-i1
   document.getElementById('dur').value=du
      }
 else {
      i2=i1+parseInt(dur)
      df=Math.floor(vd[i2]/1000000)
      mf=Math.floor((vd[i2]-(df*1000000))/10000)
      af=vd[i2]%10000
   document.getElementById('dt2').value=
           df+'/'+mf+'/'+af
   }
}
```

☞ Para você fazer

Escreva o aplicativo aqui descrito. Mostre para o professor. Atente que haverá solicitações *ad hoc* durante a avaliação. Para testar seu aplicativo, use os exemplos dados na folha, ou quaisquer outros.
Avaliação:

Depois que programar o aplicativo calcule quantos dias há entre 16/ 5/1988 e 11/ 4/1991 .

Dada a data 5/ 6/1993 qual a data obtida depois de 1568 dias?

Responda aqui:

Dias de diferença	Data achada

404-76182 - /