# Exemplos - aula 1

Site: Moodle Unicamp
Curso: T_TT007B_2020S2 - Tópicos Especiais em Telecomunicações III
Livro: Exemplos - aula 1

Impresso por: Arthur Briganti Gini 213253
Data: sexta, 9 Out 2020, 11:17

# Sumário

# 1. Exemplo 1

Escreva um programa que leia um número inteiro positivo, n, do usuário e, em seguida, exiba o soma de todos os inteiros de 1 a n. A soma dos primeiros n inteiros positivos pode ser calculado usando a fórmula:

sm = n(n+1)/2

# 2. Resposta

```
##
#  Compute the sum of the first n positive integers.
#

# Read input from the user
n = int(input("Enter a positive integer: "))

# Compute the sum
sm = n * (n + 1) / 2

# Display the result
print("The sum of the first", n, "positive integers is", sm)
```

Python includes a built-in function named `sum`. As a result, we will use a different name for our variable.

# 3. Exemplo 2

Crie um programa que leia três inteiros do usuário e os exiba de forma ordenada (do menor ao maior). Use as funções mínimo e máximo para encontrar o menor e maiores valores. O valor médio pode ser encontrado calculando a soma de todos os três valores e, em seguida, subtraindo o valor mínimo e o valor máximo.

# 4. Resposta 2

```
##
# Sort 3 values entered by the user into increasing order.
#

# Read the numbers from the user, naming them a, b and c
a = int(input("Enter the first number: "))
b = int(input("Enter the second number: "))
c = int(input("Enter the third number: "))

mn = min(a,b,c)             # the minimum value
mx = max(a,b,c)             # the maximum value
md = a + b + c - mn - mx    # the middle value

# Display the result
print("The numbers in sorted order are:")
print(" ", mn)
print(" ", md)
print(" ", mx)
```

# 5. Exemplo 3

A maioria dos anos tem 365 dias. No entanto, o tempo necessário para a Terra orbitar o Sol é um pouco mais do que isso. Como resultado, um dia extra, 29 de fevereiro, está incluído em alguns anos para corrigir essa diferença. Esses anos são chamados de anos bissextos.
As regras para determinar se um ano é ou não um ano bissexto são as seguintes:

- Qualquer ano divisível por 400 é um ano bissexto.
- Dos anos restantes, qualquer ano divisível por 100 não é um ano bissexto.
- Dos anos restantes, qualquer ano divisível por 4 é um ano bissexto.
- Todos os outros anos não são anos bissextos.

Escreva um programa que leia um ano do usuário e exiba uma mensagem indicando seja ou não um ano bissexto.

# 6. Resposta 3

```
##
#  Determine whether or not a year is a leap year.
#

# Read the year from the user
year = int(input("Enter a year: "))

# Determine if it is a leap year
if year % 400 == 0:
  isLeapYear = True
elif year % 100 == 0:
  isLeapYear = False
elif year % 4 == 0:
  isLeapYear = True
else:
  isLeapYear = False

# Display the result
if isLeapYear:
  print(year, "is a leap year.")
else:
  print(year, "is not a leap year.")
```

# 7. Exemplo 4

A duração de um mês varia de 28 a 31 dias. Neste exercício, você criará um programa que lê o nome de um mês do usuário como uma string. Então seu programa deve exibir o número de dias naquele mês. Exibir "28 ou 29 dias" para fevereiro, para que os anos bissextos sejam considerados.

# 8. Resposta 4

```
##
#  Display the number of days in a month.
#

# Read input from the user
month = input("Enter the name of a month: ")

# Compute the number of days in the month
days = 31

if month == "April" or month == "June" or \
   month == "September" or month == "November":
  days = 30
elif month == "February":
  days = "28 or 29"

# Display the result
print(month, "has", days, "days in it.")
```

# 9. Exemplo 5

Um dos primeiros exemplos conhecidos de criptografia foi usado por Júlio César. César precisava fornecer instruções escritas a seus generais, mas não queria que seus inimigos soubesse de seus planos se a mensagem caísse em suas mãos. Como resultado, ele desenvolveu o que mais tarde ficou conhecido como Cifra de César.
A ideia por trás dessa cifra é simples, cada letra da mensagem original é deslocada em 3 lugares. Como resultado, A se torna D, B se torna E, C se torna F, D torna-se G, etc. As últimas três letras do alfabeto são colocadas em volta do
começando: X torna-se A, Y torna-se B e Z torna-se C. Os caracteres que não são letras são não modificado pela cifra.
Escreva um programa que implemente uma cifra de César. Permita que o usuário forneça o mensagem e a quantidade de deslocamento e, em seguida, exiba a mensagem deslocada. Garanta que seu programa codifica letras maiúsculas e minúsculas. Seu programa deve também suporta valores de deslocamento negativos para que possa ser usado para codificar mensagens e decodificar mensagens.

# 10. Resposta 5

```
# Implement a Caesar cipher that shifts all of the letters in a message by an amount
# provided by the user. Use a negative shift value to decode a message.
#

# Read the message and shift amount from the user
message = input("Enter the message: ")
shift = int(input("Enter the shift value: "))

# Process each character, constructing a new message
new_message = ""
for ch in message:
  if ch >= "a" and ch <= "z":
    # Process a lowercase letter by determining its
    # position in the alphabet (0 - 25), computing its
    # new position, and adding it to the new message
    pos = ord(ch) - ord("a")
    pos = (pos + shift) % 26
    new_char = chr(pos + ord("a"))
    new_message = new_message + new_char
  elif ch >= "A" and ch <= "Z":
    # Process an uppercase letter by determining its position in the alphabet
    # (0 - 25), computing its new position, and adding it to the new message
    pos = ord(ch) - ord("A")
    pos = (pos + shift) % 26
    new_char = chr(pos + ord("A"))
    new_message = new_message + new_char
  else:
    # If the character is not a letter then copy it into the new message
    new_message = new_message + ch

# Display the shifted message
print("The shifted message is", new_message)
```

The `ord` function converts a character to its integer position within the ASCII table. The `chr` function returns the character from the ASCII table in the position provided.

# 11. Exemplo 6

Escreva um programa que converta um número binário (base 2) em decimal (base 10). Seu programa deve começar lendo o número binário do usuário como uma string. Então ele deve calcular o número decimal equivalente processando cada dígito no número binário. Finalmente, seu programa deve exibir o número decimal equivalente.

# 12. Resposta 6

```
##
#  Convert a number from Decimal (base 10) to Binary (base 2)
#
NEW_BASE = 2

# Read the number to convert from the user
num = int(input("Enter a non-negative integer: "))

# Generate the binary representation of num,
# storing it in result
result = ""
q = num

# Perform the body of the loop once
r = q % NEW_BASE
result = str(r) + result
q = q // NEW_BASE

# Keep on looping until q == 0
while q > 0:
  r = q % NEW_BASE
  result = str(r) + result
  q = q // NEW_BASE

# Display the result
print(num, "in Decimal is", result, "in Binary.")
```

# 13. Exemplo 7

Este exercício examina o processo de identificação do valor máximo em uma coleção de inteiros. Cada um dos inteiros será selecionado aleatoriamente a partir dos números entre
1 e 100. A coleção de inteiros pode conter valores duplicados, e alguns dos números inteiros entre 1 e 100 podem não estar presentes.

Pare um momento e pense em como você lidaria com esse problema no papel. Muitas pessoas verificariam cada inteiro em sequência e se perguntariam se o número que eles estão considerando atualmente é maior do que o maior número que eles viram anteriormente. Se for, então eles esquecem o número máximo anterior e lembram o número atual como o novo número máximo. Esta é uma abordagem razoável,
e resultará na resposta correta quando o processo for executado com cuidado. Se vocês estivessem executando esta tarefa, quantas vezes você esperaria precisar atualizar o
valor máximo e lembra de um novo número?

Embora possamos responder à pergunta feita no final do parágrafo anterior usando teoria da probabilidade, vamos explorá-la simulando a situação. Crie um programa que começa selecionando um número inteiro aleatório entre 1 e 100. Salve este inteiro como o número máximo encontrado até agora. Depois que o inteiro inicial foi selecionado, gera 99 inteiros aleatórios adicionais entre 1 e 100. Verifique cada
inteiro conforme é gerado para ver se é maior do que o número máximo encontrado tão longe. Se for, então o seu programa deve atualizar o número máximo encontrado
e conte o fato de você ter feito uma atualização. Exibir cada inteiro depois de você gerá-lo. Inclua uma notação com os inteiros que representam um novo máximo.
Depois de exibir 100 inteiros, seu programa deve exibir o máximo valor encontrado, junto com o número de vezes que o valor máximo foi atualizado durante o processo. A saída parcial do programa é mostrada abaixo, com... representando os inteiros restantes que seu programa exibirá. Execute o seu
programa várias vezes. É o número de atualizações realizadas no valor máximo o que você esperava?

```
30
74 <== Update
58
17
40
37
13
34
46
52
80 <== Update
37
97 <== Update
45
55
73
...

The maximum value found was 100
The maximum value was updated 5 times
```

# 14. Resposta 7

```
##
#  Find the maximum of 100 random integers, counting the number of times the
#  maximum value is updated during the process
#
from random import randrange

NUM_ITEMS = 100

# Generate the first number and display it
mx_value = randrange(1, NUM_ITEMS + 1)
print(mx_value)

# Count of the number of updates
num_updates = 0

# For each of the remaining numbers
for i in range(1, NUM_ITEMS):
  # Generate a new random number
  current = randrange(1, NUM_ITEMS + 1)

  # If the generated number is the largest one we have seen so far
  if current > mx_value:
    # Update the maximum and count the update
    mx_value = current
    num_updates = num_updates + 1
    # Display the number, noting that an update occurred
    print(current, "<== Update")
  else:
    # Display the number
    print(current)

# Display the summary results
print("The maximum value found was", mx_value)
print("The maximum value was updated", num_updates, "times")
```