UNIVERSIDADE FEDERAL DE SÃO JOÃO DEL-REI - UFSJ
DEPARTAMENTO DE ENGENHARIA ELÉTRICA - DEPEL
COORDENAÇÃO DE ENGENHARIA ELÉTRICA - COELE

GABRIEL AUGUSTO SILVA BATISTA

TRABALHO 1

São João del-Rei – MG
Agosto de 2023

Questão 1 –

```python
decimal = 0.0
potencia = -1

num = ['1.100001010001001', '1.100001010001001', '1.101101100101001', '1.100100111111011', '1.000111110101010']
exp = [-1, 5, 8, 17, -14]

for cont in range(len(num)):
    partes = num[cont].split('.')
    inteira = int(partes[0], 2)

    for i in partes[1]: #Percorre por cada numero da parte decimal
        if i == '1': #Se o numero for 1, entao realiza a opreacao multiplicando pelo expoente e a cada repetição diminui o expoente
            decimal += 2 ** potencia
        potencia -= 1

    res = ((inteira + decimal) * 2 **exp[cont]) # da o resultado final e multiplica pelo expoente
    print("Convertido:", res)
    decimal = 0.0 #Reseta variaveis para proxima iteracao
    potencia = -1

```

A lógica do código acima consiste em: Armazenar números binários normalizados em um vetor chamado "num" e seus respectivos expoentes no vetor "exp".

Um laço de repetição é executado até percorrer todas as posições do vetor "num". Durante cada iteração, na linha 8, o binário armazenado na posição "cont" (variável de controle) é separado em sua parte inteira e parte fracionária utilizando a função split para realizar a separação onde encontrar um "." .

Na linha 9, uma função do Python é utilizada para converter a parte inteira do binário da base 2 para a base 10, e o resultado é armazenado na variável chamada "inteira".

Na linha 11, um segundo laço de repetição é utilizado para percorrer cada bit da parte fracionária do binário. Em cada iteração, se o bit analisado for igual a 1, ele é multiplicado pela potência correspondente à sua posição: o primeiro bit é elevado a -1, o segundo a -2 e assim por diante.

Por fim, na linha 16, a parte inteira e a fracionária são combinadas em uma única variável e multiplicadas pelo expoente. O resultado é exibido na linha 17.

Resultado exibido no terminal:

```
Convertido: 0.7599029541015625
Convertido: 48.6337890625
Convertido: 438.3203125
Convertido: 206828.0
Convertido: 6.850436329841614e-05
```

Questão 2 –

```python
import numpy as np

valores = [5.21, 35.0, 47.5]
exp = [0, -95, 112]
for i in range(len(valores)):
    x = np.float64(valores[i]*(10**exp[i]))
    y = np.float32(x)
    dif = x-y
    print(f'64: {x}\n32: {y}\nDiferença: {dif}\n\n')

```

Na questão 2, utilizei a biblioteca numpy para forçar o Python a armazenar variáveis em formatos de 32 e 64 bits. Já que por padrão ele tem suporte a ambos os tipos e atribui automaticamente o tamanho necessário.

Seguindo a mesma lógica da questão 1, armazenei os números e os expoentes em seus respectivos vetores. Em seguida, um laço de repetição foi criado para percorrer os elementos do vetor "valores".

Na linha 6 peguei o valor na posição da variável de controle "i" e multipliquei pelo expoente correspondente. Utilizei a função float64 da biblioteca numpy para assegurar que o valor fosse salvo em uma variável "x" com precisão de 64 bits.

Na linha 7, realizei a conversão da variável "x" para o formato de 32 bits, utilizando a função float32 da biblioteca numpy e a salvei na variável "y", na linha 8 foi calculado a diferença para encontrar o erro. Por fim, na linha 9, exibi o resultado.

Resultado exibido no terminal:

```
64: 5.21
32: 5.210000038146973
Diferença: -3.814697269177714e-08


64: 3.5e-94
32: 0.0
Diferença: 3.5e-94


c:\Users\Gabriel\Documents\Codigos\Python\64to32.py:7: RuntimeWarning: overflow encountered in cast
  y = np.float32(x)
64: 4.7499999999999994e+113
32: inf
Diferença: -inf
```

Na primeira execução a transformação foi realizada e o erro foi de $-3.18x10^{-8}$

Na segunda execução o valor que estava sendo convertido era muito pequeno (muito próximo de 0) para a variável de 32 bits, então ficou salvo como 0

Na terceira execução o valor era muito grande para a variável de 32 bits, então foi obtido o erro de overflow e a variável de 32 bits interpretou o valor como infinito.