

RELATÓRIO EP1 OCD

Grupo: Gabriel Henry Hamamoto (11369788), Giovani Verginelli Haka (11295696), Mirela Mei (11208392)

1. Para compilar é necessário estar no diretório do EP e compilar os arquivos **métodos.java** e **converter.java**. ***A versão Java Runtime utilizada foi a 55 (Java 11)**
2. Para executar o programa, no prompt de comando, já dentro do diretório onde se encontra o arquivo `calculadora_bin.jar`, execute o seguinte código: **java -jar calculadora_bin.jar**
3. O arquivo **converter.java** contém o arquivo main, que recebe os valores pedidos, filtra as condições dos valores, utiliza as funções do arquivo **metodos.java** e retorna o resultado. O arquivo **metodos.java** contém as funções a serem utilizadas na soma, subtração, multiplicação e divisão, assim como outras funções auxiliares que fizemos.
4. Utilizamos **String** para receber os valores em binário e depois, pelo método **validarArray**, colocamos os valores recebidos em arrays, com base no número de bits e começando pelo último índice para facilitar as operações.

A **divisão** foi implementada com base na explicação encontrada no livro do Stallings (10ª edição), utilizamos os métodos **shiftLeftA** e **shiftLeftQ** para realizar os deslocamentos das operações, e o método **subtrair**, específico para subtrair na divisão.

Para identificarmos o tamanho do array necessário nos cálculos em **float**, nós criamos arrays auxiliares com o tamanho do maior número binário em bits mais a diferença do expoente dos números binários mais 1 bit, pra conseguir realizar o deslocamento da vírgula sem prejudicar os cálculos.

5. Testes ao final.

6. Nas operações de inteiros conseguimos dizer que o overflow acontece imprimindo uma mensagem na tela, mas ao mesmo tempo retornando um resultado inválido. Para underflow não conseguimos encontrar solução e não conseguimos retornar nas operações de float o valor do expoente no resultado.

Nas operações em float, conseguimos desenvolver apenas a soma de floats com binários de mesmo expoente. Não conseguimos identificar overflow e nem underflow na soma.

Quanto a soma de floats com expoentes diferentes, poderíamos resolver igualando os expoentes dos números binários, deslocando a vírgula para a esquerda e adicionando zero se necessário. A função de soma continuaria sendo a mesma que em floats de expoentes iguais. Para a subtração de float bastaria fazer o complemento de 2 do número, se preciso ajustar os expoentes e a vírgula e em seguida utilizar a função de soma para floats.

Não tivemos tempo de realizar as operações de multiplicação e divisão de float. Para multiplicação de floats, adaptaríamos o algoritmo de Booth para executar o cálculo com a presença da vírgula. Para realizar a divisão de inteiros negativos, resolveríamos calculando o complemento de 2 para o número negativo.

Na divisão, encontramos um bug que ocorre quando o número de bits que a pessoa coloca é maior do que o número em si (então o array fica com zeros a mais), um provável jeito de resolver seria colocando um índice que aponta para o último valor “válido” do array e calcula todo o resto da divisão com base nele.

7. links utilizados além do livro:

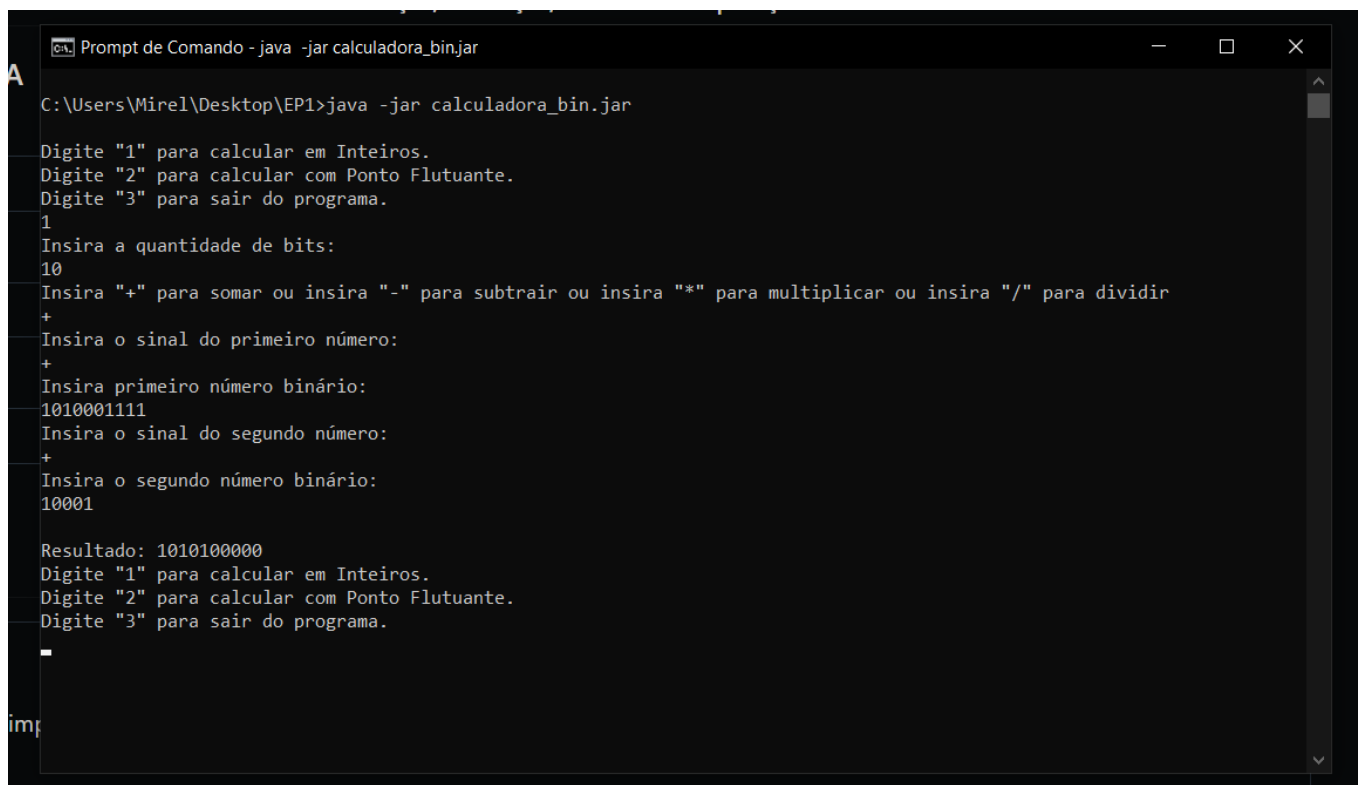
https://www.inf.pucrs.br/emoreno/undergraduate/EC/arqi/class_files/Aula08.pdf (geral)

<http://ww2.inf.ufg.br/~fmc/arqcomp/Aritimetica.pdf> (geral)

<http://www2.ufersa.edu.br/portal/view/uploads/setores/145/arquivos/arq/aulas/08%20-%20Aritm%C3%A9tica%20Computacional.pdf> (geral)

<https://sabercomlogica.com/pt/o-algoritmo-da-divisao/> (divisão)

<https://courses.cs.vt.edu/~cs1104/Division/ShiftSubtract/Shift.Subtract.html> (divisão)



```
Prompt de Comando - java -jar calculadora_bin.jar

C:\Users\Mirel\Desktop\EP1>java -jar calculadora_bin.jar

Digite "1" para calcular em Inteiros.
Digite "2" para calcular com Ponto Flutuante.
Digite "3" para sair do programa.
1
Insira a quantidade de bits:
10
Insira "+" para somar ou insira "-" para subtrair ou insira "*" para multiplicar ou insira "/" para dividir
+
Insira o sinal do primeiro número:
+
Insira primeiro número binário:
1010001111
Insira o sinal do segundo número:
+
Insira o segundo número binário:
10001

Resultado: 1010100000
Digite "1" para calcular em Inteiros.
Digite "2" para calcular com Ponto Flutuante.
Digite "3" para sair do programa.

```

```
Prompt de Comando - java -jar calculadora_bin.jar
Insira primeiro número binário:
1010001111
Insira o sinal do segundo número:
+
Insira o segundo número binário:
10001

Resultado: 1010100000
Digite "1" para calcular em Inteiros.
Digite "2" para calcular com Ponto Flutuante.
Digite "3" para sair do programa.
1
Insira a quantidade de bits:
15
Insira "+" para somar ou insira "-" para subtrair ou insira "*" para multiplicar ou insira "/" para dividir
+
Insira o sinal do primeiro número:
+
Insira primeiro número binário:
11111111111111
Insira o sinal do segundo número:
+
Insira o segundo número binário:
100001

Resultado: 1000000000100000
Digite "1" para calcular em Inteiros.
Digite "2" para calcular com Ponto Flutuante.
Digite "3" para sair do programa.
```

```
Prompt de Comando - java -jar calculadora_bin.jar
Digite "2" para calcular com Ponto Flutuante.
Digite "3" para sair do programa.
3
C:\Users\Mirel\Desktop\EP1>java -jar calculadora_bin.jar

Digite "1" para calcular em Inteiros.
Digite "2" para calcular com Ponto Flutuante.
Digite "3" para sair do programa.
2
1
Insira a quantidade de bits:
10
Insira "+" para somar ou insira "-" para subtrair ou insira "*" para multiplicar ou insira "/" para dividir
*
Insira o sinal do primeiro número:
+
Insira primeiro número binário:
10100
Insira o sinal do segundo número:
+
Insira o segundo número binário:
101
Overflow
Overflow

Resultado: 00000000000001100100
Digite "1" para calcular em Inteiros.
Digite "2" para calcular com Ponto Flutuante.
Digite "3" para sair do programa.
```