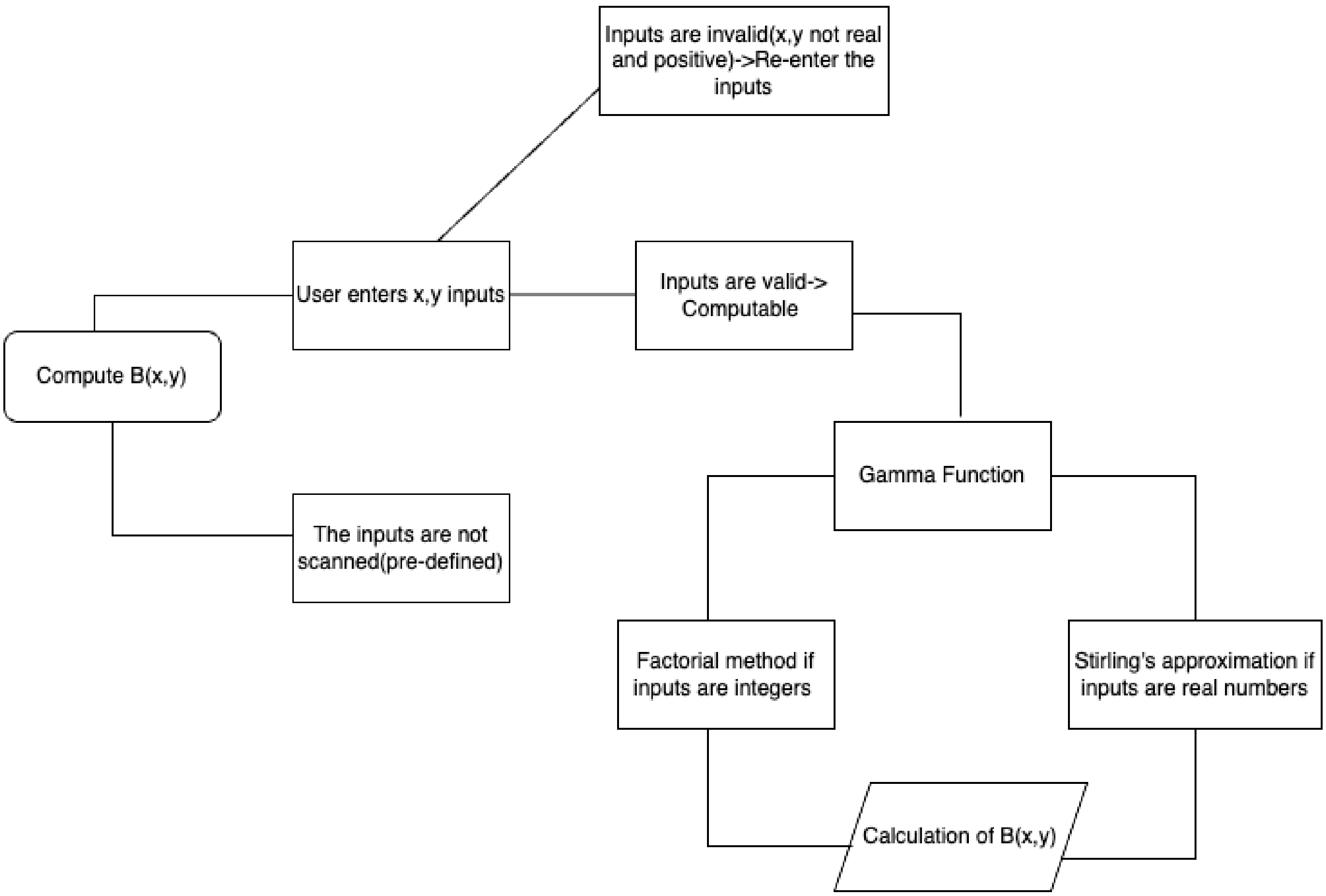
SOEN6011 Project: Problem-3

Function: B(x,y) 40197897-Pal Patel

3.1 Mind Map Beta Calculator



Algorithm 1- Stirling's Approximation

Stirling's approximation is an approximation of factorials. It gives an accurate value of factorials even when the value of the number is small.The formula gives an approximate answer to the factorials included in the gamma function Γx [?].

The factorial form of the Beta function is equivalent in terms of the Gamma function as follows: $B(x, y) = \frac{\Gamma x \Gamma y}{\Gamma(x+y)}$

Here, $\Gamma x = \sqrt{\frac{2\pi}{x}}(\frac{x}{e})^x$

Advantages:

- Stirling's approximation gives accurate answers even for small numbers.
- Stirling's approximation works for all positive real numbers R+.

Disadvantages:

- When the numbers are integers and not real numbers, the accuracy of Stirling's algorithm is less than that of the corresponding factorial method.
- Implementation of Stirling's algorithm is more complex as compared to the Factorial method. The factorial method takes in integers and simply computes the factorials and calculates the final results. Whereas in the case of Stirling's algorithm, the square root and power of the numbers are to be calculated computationally.

Technical reasons for selecting this algorithm:

- It can be useful for all the valid input values to the Beta function where validity means that the inputs are real and greater than 0.
- Stirling's approximation algorithm works accurately even for small numbers.

Algorithm 1 Stirling's Approximation for calculating the Beta Function

```
procedure SQUAREROOT(x)
    temp ← 0.0
    root ← x/2
    temp = root
    root = (temp + (x/temp))/2 ((temp − root)! = 0)
    return root                                  ▷ returns the square root of x
end procedure

procedure POWER(a,b)
    powr ← 1
    i = 1
    while  doi <= b
        powr = powr ∗ a

        return powr                              ▷ Returns a to the power b


        procedure GAMMA(a)
            a ← a − 1
            pi ← 3.141592653589793
            e ← 2.718281828459045
            root ← 2 ∗ pi ∗ x
            sqr ← SQUAREROOT(root)
            pow ← POWER(a, e)
            result ← (sqr) ∗ (pow)
            return gamma                         ▷ Returns the gamma values
        end procedure

        procedure CALCULATEBETA(a, b)
            gamma_a ← GAMMA(a)
            gamma_b ← GAMMA(b)
            gamma_c ← GAMMA(a + b)
            beta ← (gamma_a ∗ gamma_b) / gamma_c
            return beta                          ▷ Returns the output of the beta function
        end procedure

        beta ← CALCULATEBETA(x, y)
```

Algorithm 2-Beta function using factorial

This algorithm is application to only the integers(pre-requisite is that it be greater than 0)

$B(x, y) = \frac{\Gamma x \Gamma y}{\Gamma(x+y)}$

where, $\Gamma x = (x-1)!$
$\Gamma y = (y-1)!$
$\Gamma x + y = (x+y-1)!$

Advantages:

- It has more accurate answers than Stirling's approximation when Stirling's approximation algorithm is applied to integers.
- This algorithm does not require the calculation of square roots, or power. The computations are simpler as only the factorials are to be calculated.

Disadvantages:

- The algorithm works for only integers.
- Because finding the factorial involves recursion, there are possible chances of the StackOverFlow exception when the numbers in the input are large.

Technical reasons for selecting this algorithm:

- Considering the case where we want to compute the beta function for integers, the factorial method would give the most quick and accurate results.
- The implementation is simple and less complex as compared to Stirling's approximation.

Algorithm 2 Factorial for calculating the Beta Function

```
procedure FACTORIAL(temp)
    if temp ≤ 1 then
        return 1
    else
        return temp * getFactorial(temp − 1)
    end if
end procedure

procedure GAMMA(x1)
    x1 ← x1 − 1
    gammavalue ← GETFACTORIAL(x1)
    return gammavalue
end procedure

procedure CALCULATEBETA(x, y)
    z ← x + y
    gammax ← GAMMA(x)
    gammay ← GAMMA(y)
    gammaz ← GAMMA(z)
    beta ← (gammax*gammay) / gammaz
    return beta                                  ▷ Final Beta Value
end procedure

beta ← CALCULATEBETA(x, y) =0
```

Bibliography

[1] stirling https://en.wikipedia.org/wiki/Stirling%27s_approximation