|  |  |
| --- | --- |
| Customer | Algorithms and programing l |
| Users | * People who need the bisection method |
| Functional requirements | * R1 Menu * R2 Select a possible function * R3 Enter a and b value * R4 Check a or b are not a root * R5 Check that exist a root on the interval * R6 Do bisection method with the f(c)>epsilon criterion * R7: Do the Cos(x) * R8: Do the Factorial * R9: Do the Pow * R10: Do the absolute value |
| Problem context | The system has to show a menu about the bisection method, request to the user a function with two values (a and b). Then check if this are not a root, that exist a root on the interval, do the bisection method with the epsilon criterion, using the own cosine of the system to calculate all the f(x) that this going to use, the cosine modify the x value to the range [-pi, pi] to operate faster but with the same result of the original x. |
| Nonfunctional requirements | * The system can’t import any library of java * The system has to be working in language java * The system has to have efficiency, the response time has to be short. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id and name | R1: Menu | | | |
| Summary | *The system shows a menu with 2 options: 1. bisection method or 0. Exit. The user enters options that repeats until the option is 0 and the menu ends.* | | | |
| Input | **Input name** | **datatype** | | **Conditions valid values** |
| option | int | | 1. *Bisection method* 2. *Exit.* |
|  |  | |  |
| Result | If the option is equals to 1, the system show the menu do the actions, and show again the menu. If is 0 the program ends. | | | |
| Output | **Output name** | | **datatype** | **Format** |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id and name | R2: Select a possible function | | | |
| Summary | *The system shows the 3 possible equation.*  *1. f(x)=2cos(x^2)*  *2. f(x)= 3x^3+7x^2+5*  *3. f(x)= x cos(x)*  *And the user enter the number for his function, saving the function to the bisection.* | | | |
| Input | **Input name** | **datatype** | | **Conditions valid values** |
| function | int | | *1. f(x)=2cos(x^2)*  *2. f(x)= 3x^3+7x^2+5*  *3. f(x)= x cos(x)* |
|  |  | |  |
| Result | The system save the function to use in the bisection | | | |
| Output | **Output name** | | **datatype** | **Format** |
| confirmationFunction | | String | “You select ” + function + “ function” |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id and name | R3: Enter a and b value | | | |
| Summary | *The system request the value a and b to the user, he enters it and if are correctly enter (Are different values) the system do and shows the values a and b for the enter function.* | | | |
| Input | **Input name** | **datatype** | | **Conditions valid values** |
| valueA | double | |  |
| valueB | double | |  |
| Result | The system do and show the values of A and B in the enter function | | | |
| Output | **Output name** | | **datatype** | **Format** |
| functionA | | double |  |
| functionB | | double |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id and name | R4: Check a or b are not a root | | | |
| Summary | *The system do f(a) and f(b) but if one of this is equals to 0, the system show that a or b are equals to a root and shows again the menu, but if isn’t it continues with the bisection.* | | | |
| Input | **Input name** | **datatype** | | **Conditions valid values** |
|  |  | |  |
|  |  | |  |
| Result | If f(a) or f(b) was equals to zero print *“the value of (a or b) match with a Zero” but if is not the system continue with the bisection* | | | |
| Output | **Output name** | | **datatype** | **Format** |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id and name | R5: Check that the interval is useful for the bisection | | | |
| Summary | *The system do f(a) \* f(b) and if the result is < 0, start to do the bisection but if is not, the system shows a message of alert and go again to the menu* | | | |
| Input | **Input name** | **datatype** | | **Conditions valid values** |
|  |  | |  |
|  |  | |  |
| Result | If f(a)\*f(b) < 0, the system continues with the bisection, but if isn’t it prints “don’t comply with f(a)\*f(b) < 0 and therefore the bisection can’t be done” and shows the menu. | | | |
| Output | **Output name** | | **datatype** | **Format** |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id and name | R6: Do bisection method with the f(c)>epsilon criterion | | | |
| Summary | *The system creates the mid-point(c) and select the interval where is the root in a loop. The iterations stops if absoluteValue(f(c))>epsilon is false, the program found the root and it shows the root with the amount of C and the f(c).* | | | |
| Input | **Input name** | **datatype** | | **Conditions valid values** |
|  |  | |  |
|  |  | |  |
| Result | *If absoluteValue(f(c))>epsilon are false the program found the root and it shows the root with the amount of C and the f(c).* | | | |
| Output | **Output name** | | **datatype** | **Format** |
| valueC | | double |  |
| numbersC | | int | NumbersC => 0 |
|  | |  |  |
|  | |  |  |
|  | |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id and name | R7: Do the Cos(x) | | | |
| Summary | *To do the different functions of the bisection, the system needs use the Cos(x) without any java library, for that reason the system need use a summation that can get the cos value. The system does the summation for each case the user enter a function and a “x” parameter that use it. The cos method modifies the x value to the range [-pi, pi] to operate faster, but with the same result of the original x.* | | | |
| Input | **Input name** | **datatype** | | **Conditions valid values** |
|  |  | |  |
|  |  | |  |
| Result | The system returns the summation to get the cos value for the value in the function that the user enters but in range [-pi, pi] with the same result. | | | |
| Output | **Output name** | | **datatype** | **Format** |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id and name | R8: Do the Factorial | | | |
| Summary | *The system gets the base of the factorial, then use a for loop until have the multiplication of all the terms of the factorial and return it.* | | | |
| Input | **Input name** | **datatype** | | **Conditions valid values** |
|  |  | |  |
|  |  | |  |
| Result | The system returns the value of the factorial multiplication of the base. | | | |
| Output | **Output name** | | **datatype** | **Format** |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id and name | R9: Do the pow | | | |
| Summary | *The system gets the base and index of the pow, and do a loop where the base is multiply by the base until the index is satisfied, and then returns the pow value* | | | |
| Input | **Input name** | **Datatype** | | **Conditions valid values** |
|  |  | |  |
|  |  | |  |
| Result | The system returns the pow value | | | |
| Output | **Output name** | | **Datatype** | **Format** |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id and name | R10: Do the absolute value | | | |
| Summary | *If the value of x that enter in the absolute value is < 0, this multiply by -1 the number to change it to a positive one, else if is a positive one don’t change. Then the system returns the value of x* | | | |
| Input | **Input name** | **Datatype** | | **Conditions valid values** |
|  |  | |  |
|  |  | |  |
| Result | If x was negative, the system return the new positive and else if was positive return the same one. | | | |
| Output | **Output name** | | **Datatype** | **Format** |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |