Babylonian method for square root

Algorithm:
This method can be derived from (but predates) Newton–Raphson method.

```
1 Start with an arbitrary positive start value x (the closer to the
   root, the better).
2 Initialize y = 1.
3. Do following until desired approximation is achieved.
  a) Get the next approximation for root using average of x and y
  b) Set y = n/x
```

Implementation:

```
/*Returns the square root of n. Note that the function */
float squareRoot(float n)
{
  /*We are using n itself as initial approximation
   This can definitely be improved */
  float x = n;
  float y = 1;
  float e = 0.000001; /* e decides the accuracy level*/
  while(x - y > e)
  {
    x = (x + y)/2;
    y = n/x;
  }
  return x;
}

/* Driver program to test above function*/
```


Interview Experiences
Advanced Data Structures
Dynamic Programming
Greedy Algorithms
Backtracking
Pattern Searching
Divide & Conquer
Mathematical Algorithms
Recursion

```
int main()
{
  int n = 50;
  printf ("Square root of %d is %f", n, squareRoot(n));
  getchar();
}
```

Example:

```
n = 4 /*n itself is used for initial approximation*/
Initialize x = 4, y = 1
Next Approximation x = (x + y)/2 (= 2.500000),
y = n/x  (=1.600000)
Next Approximation x = 2.050000,
y = 1.951220
Next Approximation x = 2.000610,
y = 1.999390
Next Approximation x = 2.000000,
y = 2.000000
Terminate as (x - y) > e now.
```

If we are sure that n is a perfect square, then we can use following method. The method can go in infinite loop for non-perfect-square numbers. For example, for 3 the below while loop will never terminate.

```
/*Returns the square root of n. Note that the function
  will not work for numbers which are not perfect squares*/
unsigned int squareRoot(int n)
{
  int x = n;
  int y = 1;
  while(x > y)
  {
    x = (x + y)/2;
    y = n/x;
  }
  return x;
}

/* Driver program to test above function*/
int main()
{
```

Geometric Algorithms

Popular Posts

All permutations of a given string
Memory Layout of C Programs
Understanding “extern” keyword in C
Median of two sorted arrays
Tree traversal without recursion and without stack!
Structure Member Alignment, Padding and Data Packing
Intersection point of two Linked Lists
Lowest Common Ancestor in a BST.
Check if a binary tree is BST or not
Sorted Linked List to Balanced BST

```
  int n = 49;
  printf (" root of %d is %d", n, squareRoot(n));
  getchar();
}
```

References;
http://en.wikipedia.org/wiki/Square_root
http://en.wikipedia.org/wiki/Babylonian_method#Babylonian_method

Asked by Snehal

Please write comments if you find any bug in the above program/algorithm, or if you want to share more information about Babylonian method.

Related Tpoics:

- Backtracking | Set 8 (Solving Cryptarithmetic Puzzles)
- Tail Recursion
- Find if two rectangles overlap
- Analysis of Algorithm | Set 4 (Solving Recurrences)

- Print all possible paths from top left to bottom right of a mXn matrix
- Generate all unique partitions of an integer
- Russian Peasant Multiplication
- Closest Pair of Points | O(nlogn) Implementation

Writing code in comment? Please use **ideone.com** and share the link here.

6 Comments GeeksforGeeks

Sort by Newest

Join the discussion…

wishall • 4 months ago

what if no.entered is between 0 & 1

abhishek08aug • a year ago

> http://en.wikipedia.org/wiki/Square_root

Vipul • 2 years ago

We can use Newton method also for this.

See http://en.wikipedia.org/wiki/N...

Na/|/|LE$$ • 2 years ago

Recent Comments

Abhi You live US or India?

Google (Mountain View) interview · 17 minutes ago

Aman Hi, Why arent we checking for conditions...

Write a C program to Delete a Tree. · 57 minutes ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 1 hour ago

Sanjay Agarwal bool tree::Root_to_leaf_path_given_sum(tree...

Root to leaf path sum equal to a given number · 1 hour ago

GOPI GOPINATH @admin Highlight this sentence "We can easily...

Count trailing zeroes in factorial of a number · 1 hour ago

newCoder3006 If the array contains negative numbers also. We...

Find subarray with given sum · 1 hour ago

1

divyaC • 4 years ago

```
float SqrRoot(int k){
   float g1,g2,temp;
   g2=k/2;

   while(fabs(g1-g2)>= 0.0001){

       if(g2*g2==k){
           break;
       }
       else{
           temp=g2;
           g2=(temp+(k/temp))/2;
           g1=temp;
       }
   }
   return g2;
```

mygaurav ↗ divyaC • a year ago

The code for a perfect square should also include the critical test case would go out of bounds....

So the code shud be like this :

int sqrt(int n) {
int x = n;
int y = 1;

```
if(x==INT_MAX)
{
x=(x-1)/2 +1;
y=n/x;
}

while(x > y)
{
x = (x + y)/2;
y = n/x;
}
return x;

}
```