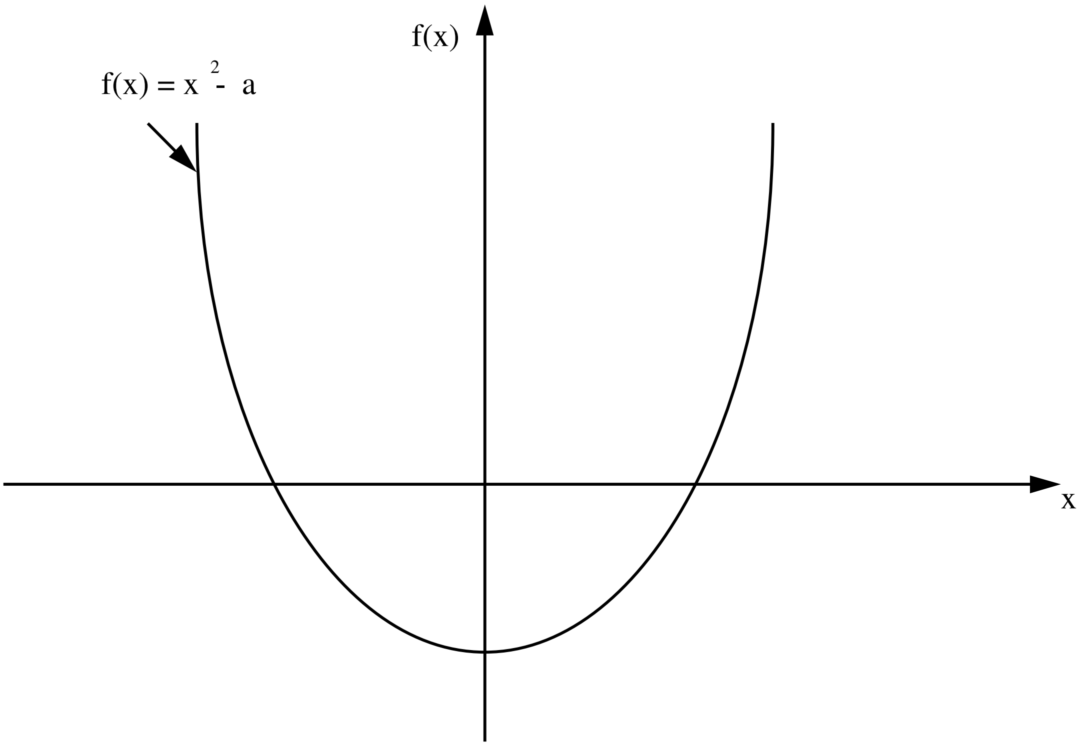
To compute the square root of a number



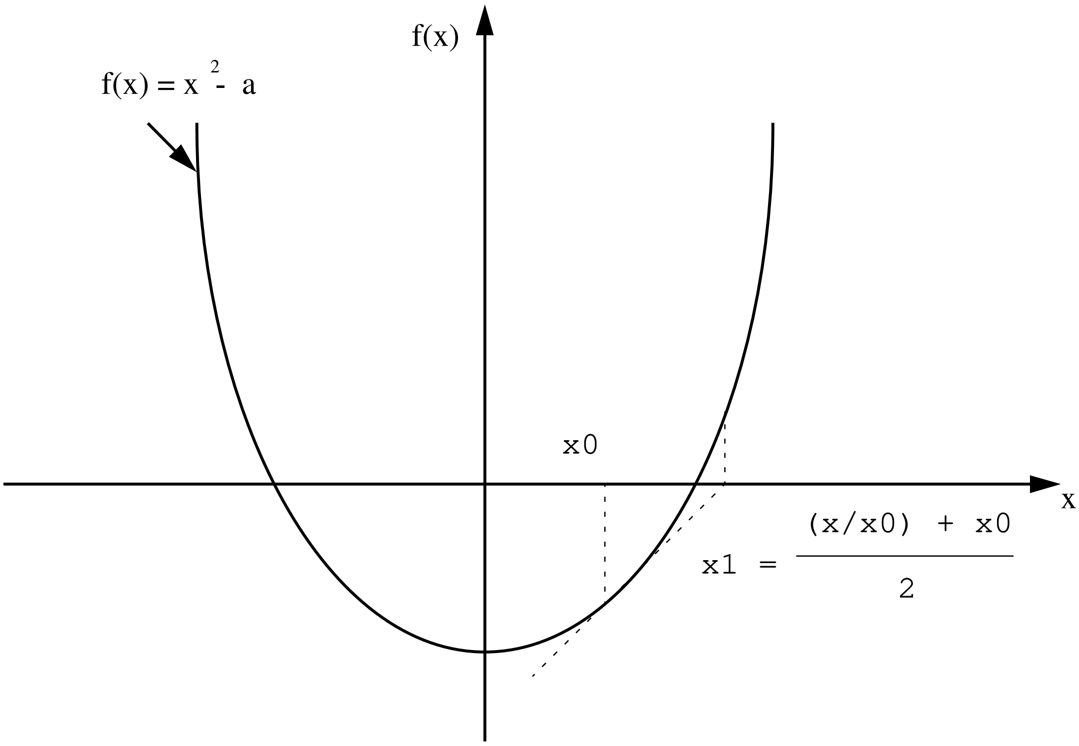
Initial Algorithm:

Square_root
begin

1. *Get the value of x for which the square root is to be found*;
2. *Make an arbitrary guess*;
3. *Improve the guess until it is close enough to the solution;*
4. Output the result;

end Square_root

Compute square root Using Newton's Method



Refine Step 3:

```
begin

  while guess is not close_enough to the solution
      repeat
        compute an improved_guess as the average
        of the current guess and (x/guess).
  end-while

end
```

Compute square root using Newton's Method (Cont'd)

```
close_enough
Input: guess, x
Output: true/false

begin

  if  abs(guess  - x) is less than some
        predetermined tolerance
    then  return true;
    else  return false;
  end-if

end close_enough


improved_guess
Input: guess, x
Output: new guess

begin
      Average ( guess, (x/guess);
end improved_guess


average
Input: two values x & y
Output: average of x and y

begin
      compute (x + y) / 2;
end average
```

C++ program for computing the square root of a number

```
#include <iostream.h>
#include <math.h>

float average(float, float);
int close_enough(float, float);
float improved_guess(float, float);

void main() {
	float guess, x;

	cout << "Enter x: ";
	cin >> x;


	guess = 1;

	while ( ! close_enough(guess, x) ) {
		guess = improved_guess(guess, x);
	}

	cout << "The square root of  " << x
		<< " is " << guess << endl;
}
```

```
int  close_enough(float guess, float x) {
      const float tol = 0.001;

      if ( fabs( guess*guess - x) < tol)
          return 1;
      else
          return 0;
}



float average(float x, float y) {
      return (x + y) / 2;
}



float improved_guess(float guess, float x) {
      return average(guess, (x/guess));
}
```