After an "Hour of Code" now what?

2016 Curtis Center Mathematics and Teaching Conference

Chris Anderson
Professor and Director of the Program in Computing
UCLA Dept. of Mathematics

March 5, 2016
New push in K-12 education: “Computer Science For All”

On Sunday, The Simpsons declared computer coding class the nation’s latest educational fad (script). Proving Principal Skinner’s point, K12CS.org on Thursday announced a New Framework to Define K-12 Computer Science Education, the collaboration of participants from a number of states (MD, CA, IN, IA, AR, UT, ID, NE, GA, VA), large school districts (NYC, Chicago, San Francisco), technology companies (Microsoft, Google, Apple), organizations (Coda.org, ACM, CSTA, ISTE, MOSSCAN, CSNYC), and individuals (higher ed faculty, researchers, K-12 teachers, and administrators). “A steering committee initially comprised of the Computer Science Teachers Association, the Association for Computing Machinery, and tech bantzelled and led Code.org will oversee this project,” explained a CSTA blog post. “Funding for the project will be provided by Code.org and the ACM. The framework will identify key K-12 computer science concepts and practices we expect students exiting grades 2, 5, 8, and 12 to know.”

In a FAQ, K12CS.org envisions a Programming and Algorithm standard for 1st Graders that calls for the 5-year-olds to "Work collaboratively in clear roles (e.g., pair programming) to construct a problem solution of a sequence of book-based programming commands." A day before the announcement, Politico reported that K-12 CS education is expected to get a State of the Union mention this year, and that the White House and U.S. Dept. of Education have been touting for CS success stories in conjunction with the announcement of a broad set of new commitments to CS Education in early 2016.


---

[Link](http://developers.slashdot.org/story/16/01/08/1446250)
“Computer Science For All” : How will this work?

http://www.npr.org/sections/ed/2016/01/12/462698966

http://developers.slashdot.org/story/16/01/17/0548234/
The Hour of Code is a global movement reaching tens of millions of students in 180+ countries. Anyone, anywhere can organize an Hour of Code event. One-hour tutorials are available in over 40 languages. No experience needed. Ages 4 to 104.
After the *Hour of Code: now what?*

• Meaningful assignments accessible to K-12 students?

How about:

*Assignments that involve getting the computer to do mathematics.*

Why will this work?
Why this will work...

One of the things computers are designed to do:

**Arithmetic very, very, very, very fast**

In the context of K-12 teaching:

**Middle school math calculations very, very, very, very fast**
Why this will work...

One of the things computers are designed to do:

**Arithmetic very, very, very, very fast**

In the context of K-12 teaching:

**Middle school math calculations very, very, very, very fast**

How fast?

This laptop = 1.8 billion arithmetic operations per second (1 core)
  = 7.2 billion arithmetic operations per second (4 cores)

-- or --

This laptop = 1.8 billion MSM calculations per second (1 core)
  = 7.2 billion MSM calculations per second (4 cores)

MSM = Middle School Math
Why this will work...

Students know enough math by middle school to use this computational power!!!

The barrier?

They can't program ... but when that changes ...
Some examples

First assignments

Computing $\pi$

Solving equations (finding roots)

Computing $\sqrt{b}$
Early programming assignments usually involve math

“Around the Track” Programming Assignment (usually #2):

Running once around the track is 1/4 of a mile.

Write a program that asks the user to input

- the number of laps he/she has run in minutes
- the total time taken

and then computes and prints out his/her speed in miles per hour.
First programming assignments usually involve math

Code (C++)

```cpp
#include <iostream>
using namespace std;

int main()
{
    double milesPerLap = 0.25;
    double hoursPerMinute = 1.0/60.0;

    double laps;
    double time;
    double speed;

    cout << "Enter laps : ";
    cin >> laps;
    cout << "Enter time (minutes) : ";
    cin >> time;

    speed = (laps*milesPerLap)/(time*hoursPerMinute);
    cout << "Speed : " << speed << " miles per hour" << endl;
}
```

Typical input and output

Enter laps : 2  
Enter time (minutes) : 8  
Speed : 3.75 miles per hour  

The “math” part  \( d = r \times t \) (dirt)

speed = (laps*milesPerLap)/(time*hoursPerMinute);
Around the track: As a programming assignment

What this “exercises” as programming language assignment

- Console input/output <iostream>
- Order of operations
- Fundamental data types (integers vs. reals)
- Integer vs real arithmetic operations

*It's an appropriate programming language assignment*

*Middle school math calculations*
Later programming assignments?

The following examples demonstrate what you can get by combining

- a few additional language constructs
- middle school math calculations
Computing \( \pi \)

\[ \pi = 2 \times \text{Area of} \]

radius = 1

Programming Assignment:

Write a program to approximate \( \pi \) by 2 times the sum of the areas of N rectangles that approximate a half-circle of radius 1.

Use your program to determine how large N has to be to get \( 3.14159 \).
Computing $\Pi$ : Many possibilities

Time to be organized!
Tip: Use rectangles with the same size base. The picture gives suggested rectangles to use.

Write a program that adds up N of these rectangles and then multiples the result by 2.

*Student:* “But they don't fit inside the circle!”

*CS Instructor:* “Nobody said they have to, just try it and see what happens. Or make them fit inside and see what happens.”

*Student:* “But how do I figure out the area of each rectangle?”

*CS Instructor:* “Work it out. I’ve talked to your math teacher, and you should be able to do this. If you get it wrong, then you’ll know soon enough, and you’ll fix it.”
Computing $\pi$: Algorithm details

- Divide up $[-1, 1]$ into $N$ panels for the bases of each rectangle.
- As the height, use the point on the circle above the midpoint of each panel.

The formulas: Given or derived by the student

$$\pi \approx 2 \times \text{Approximate area}$$

$$x_i = \frac{2i}{N} - 1$$

$$\text{Area} \approx \sum_{i=1}^{i=N} \left(1 - \left(\frac{2i - 1}{N} - 1\right)^2\right) \left(\frac{2}{N}\right)$$
Computing \( \pi \): Computer code

Computer code (C++):

```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main()
{
    long N;
    double area;
    double approxPi;

    cout << "Enter the number of rectangles N : " << endl;
    cin >> N;

    area = 0.0;
    for(long i = 1; i <= N; i++)
    {
        area += sqrt(1.0 - ((2.0*i - 1.0)/N -1.0)*((2.0*i - 1.0)/N -1.0))*(2.0/N);
    }

    approxPi = 2.0*area;

    cout.precision(14);

    cout << "Results using " << N << " Rectangles " << endl;
    cout << "Approximate value of PI : " << approxPi << endl << endl;
    cout << "XXXXX Program Complete XXXX" << endl;
}
```
Computing $\pi$: Computer code

The important “math” part

```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main()
{
    long           N;
    double      area;
    double  approxPi;

    cout << "Enter the number of rectangles N : " << endl;
    cin  >> N;

    area = 0.0;
    for(long i = 1; i <= N; i++)
    {
        area += sqrt(1.0 - ((2.0*i - 1.0)/N -1.0)*((2.0*i - 1.0)/N -1.0))*(2.0/N);
    }

    approxPi = 2.0*area;

    cout.precision(14);

    cout << "Results using " << N << " Rectangles " << endl;
    cout << "Approximate value of PI : " << approxPi << endl << endl;
    cout << "XXXX Program Complete XXXX" << endl;
}
```
The important “math” part still involves middle school math calculations:

```java
area = 0.0;
for (long i = 1; i <= N; i++)
{
    area += sqrt(1.0 - ((2.0*i - 1.0)/N -1.0)*((2.0*i - 1.0)/N -1.0))*(2.0/N);
}
```

and is easily expressed in most programming languages:

C++, C, Java, Java Script, Python, Pascal, Fortran, C#, PHP, Perl, Matlab, ...

*The assignment makes sense in whatever language “they” choose.*
Computing $\pi$: As a programming assignment

What this “exercises” as programming language assignment

- For loops (flow control)
- Mathematical function library `<cmath>`
- Console input/output `<iostream>`
- Order of operations
- Fundamental data types (integers vs. reals)
- Integer vs real arithmetic operations

*It's not much more than an early programming assignment*
Seeking N so the approximation starts with 3.14159

Enter the number of rectangles N:
1000
Results using 1000 Rectangles
Approximate value of PI: 3.1416234568199

Enter the number of rectangles N:
2000
Results using 2000 Rectangles
Approximate value of PI: 3.1416035449129

Enter the number of rectangles N:
3000
Results using 3000 Rectangles
Approximate value of PI: 3.1415985822088

N = 2601 to get 3.1415999974076

Found using “bisection” search

Enter the number of rectangles N:
2601
Results using 2601 Rectangles
Approximate value of PI: 3.1415999974076

Cool. It can do a billion operations pretty fast, so let’s try N = 1 billion.

Hmmm. Why isn't it better? Finite precision effects?

3.141592653589793238462...
Computing $\pi$: A more accessible assignment?

The Magic Pi Program

Write a program that:

- Prompts you for an integer $N$
- Adds up and then prints out the sum of the following $N$ numbers:

$$\sum_{i=1}^{N} \sqrt{1 - \left(\frac{2i-1}{N} - 1\right)^2} \left[ \frac{4}{N} \right]$$

How big does $N$ have to be to get a “good” $\pi$?
Computing $\pi$: They will still need to know some math

Making sense of what goes wrong....

Loop error: N+1 not N

```c
area = 0.0;
for(long i = 1; i <= N+1; i++)
{
    area  += sqrt(1.0-((2.0*i-1.0)/N-1.0)*((2.0*i-1.0)/N-1.0))*(2.0/N);
}
```

Enter the number of rectangles N:
1000
Results using 1000 Rectangles
Approximate value of PI: -nan

The last calculation in the loop evaluates
$$\sqrt{-\frac{1}{N}}$$

They need to know that taking the square root of a negative number is not such a good idea ...

nan = “not a number”
Some examples

First assignments

Computing $\pi$

- Solving equations (finding roots)

Computing $\sqrt{b}$
Solving equations

Background the students would need to be reminded of

- Solving equations is the same as finding roots.

Example: Finding \( x \) so that

\[
\begin{align*}
x^5 - 5x &= -1
\end{align*}
\]

Is the same as finding roots of

\[
\begin{align*}
x^5 - 5x + 1 &= 0
\end{align*}
\]

- Finding roots of \( f(x) \) is the same as finding where the graph of \( f(x) \) hits the x-axis
Solving equations

Programming assignment:

Create a program that uses the computer to find where the graph of $f(x)$ intersects the x-axis -- the value of $x$ so that $f(x) = 0$. 
Solving equations

An algorithm:

“The next guess is the root of the line through \((x_{n-1}, f(x_{n-1}))\) and \((x_n, f(x_n))\)”

Discard \(x_{n-1}\)
Set \(x_{n+1} = \) next guess
Use \((x_n, f(x_n))\) and \((x_{n+1}, f(x_{n+1}))\)

Repeat
After some thought and some middle school math:

This method is known as the “secant” method.
Solving equations: Demonstration screen capture
Solving equations: They've done something amazing

They've written a program to find real roots of any polynomial equation!

Solution by “formulas”

Linear equations \( x + \ldots = 0 \)

Quadratic equations \( x^2 + \ldots = 0 \)

Cubic equations \( x^3 + \ldots = 0 \)

Quartic equations \( x^4 + \ldots = 0 \)

Solution with a computer

Linear equations

+ middle school math

+ computer

A non-solvable quintic example
(Wikipedia: solvability by radicals)

\( x^5 - x - 1 = 0 \)

\( \cos(x) + x^3 + x = 0 \) too!
Solving equations: One has to be careful

This method doesn't always work.

Example: Failure can happen with initial guesses that “aren't good”

\[ f(x) = xe^{-x} \quad x_0 = 1 \quad x_1 = 2 \]
Some examples

First assignments

Computing $\pi$

Solving equations (finding roots)

- Computing $\sqrt{b}$
Solving equations : A more accessible assignment?

Find roots of a specific equation.

\[ x^2 - b = 0 \]

Why is this an interesting thing to do?
Solving equations: A more accessible assignment?

Find roots of a specific equation.

\[ x^2 - b = 0 \]

Why is this an interesting thing to do?

...because you get \( \sqrt{b} \) for the positive root.

It's a clever way to compute square roots!
Computing $\sqrt{b}$: Details

Computing square roots:

$$x = \sqrt{b}$$ is a root of the equation $x^2 - b = 0$

Idea: Compute the $\sqrt{b}$ by using the secant method to find the roots of $x^2 - b = 0$.

The secant method recurrence for $f(x) = x^2 - b$ is

$$x_{n+1} = x_n - \left( \frac{x_n^2 - b}{x_n + x_{n-1}} \right)$$

Another iteration to try:

$$x_{n+1} = x_n - \left( \frac{x_n^2 - b}{2x_n} \right)$$
Finding square roots

The Root Digging Program

Write a program to compute $\sqrt{b}$ using the following recurrence:

$$x_{n+1} = x_n - \left( \frac{x_n^2 - b}{x_n + x_{n-1}} \right) \quad x_0 = 0 \quad x_1 = b$$

Print out how close $x_n$ is to $\sqrt{b}$ at each iteration.

Extra credit: Compare your results with the results obtained with the following recurrence:

$$x_{n+1} = x_n - \left( \frac{x_n^2 - b}{2x_n} \right) \quad x_0 = b$$
Square roots

Computer Code (C++)

```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main()
{
    long N;
    double b;
    double xNm1, xN, xStar;

    cout << "Enter the value whose square root you seek : " << endl;
    cin  >> b;
    cout << "Enter the number of iterations N : " << endl;
    cin  >> N;
    cout.precision(14);
    xNm1 = 0.0;
    xN = b;
    for(long i = 1; i <= N; i++)
    {
        xStar = xN - (xN*xN - b)/(xN + xNm1);
        xNm1 = xN;
        xN = xStar;
        cout << i << " : Approximation " << abs(xN)
             << " Error : " << abs(sqrt(b) - abs(xN)) << endl;
    }
    cout << "XXXX Program Complete XXXX" << endl;
}
```
Square roots : Demonstration results (secant method)

Enter the value whose square root you seek :
2.0
Enter the number of iterations N :
10

1 : Approximation 1               Error : 0.4142135623731
2 : Approximation 1.3333333333333 Error : 0.080880229039762
3 : Approximation 1.4285714285714 Error : 0.014357866198333
4 : Approximation 1.4137931034483 Error : 0.00042045892481934
5 : Approximation 1.414214384749  Error : 2.1238982250704e-06
6 : Approximation 1.4142135626889 Error : 3.157745176452e-10
7 : Approximation 1.4142135623731 Error : 4.4408920985006e-16
8 : Approximation 1.4142135623731 Error : 2.2204460492503e-16
9 : Approximation 1.4142135623731 Error : 0
10 : Approximation 1.4142135623731 Error : 2.2204460492503e-16

Enter the value whose square root you seek :
1234.0
Enter the number of iterations N :
15

1 : Approximation 1               Error : 34.128336140501
2 : Approximation 1.9983805668016 Error : 33.129955573699
3 : Approximation 412.22198217661 Error : 377.09364603611
4 : Approximation 4.9678301297439 Error : 30.160506010757
5 : Approximation 7.8665602236465 Error : 27.261775916854
6 : Approximation 99.19284826491  Error : 64.064512124409
7 : Approximation 18.814848158313 Error : 16.313487982188
8 : Approximation 26.272001508918 Error : 8.8563346315828
9 : Approximation 38.332767358137 Error : 3.2044312176366
10 : Approximation 34.689057188421 Error : 0.43927895207985
11 : Approximation 35.109059175072 Error : 0.019276965428297
12 : Approximation 35.128457461327 Error : 0.0001213208266088
13 : Approximation 35.128336107204 Error : 3.3296984724984e-08
14 : Approximation 35.128336140501 Error : 5.6843418860808e-14
15 : Approximation 35.128336140501 Error : 0

XXXX Program Complete XXXX
# Math + Computers + Programming?

## Math assignments with programming

**In the Math classroom**
- Using calculators
- Using spreadsheets
- Using Math Apps/Demonstrations
  - *
  - *
  - *

Success doesn't depend on understanding how the computer is getting things done.

Programming “doesn't get in the way”

## Programming assignments with math

**In the CS classroom**
- **First assignments**
- Computing $\pi$
- Solving equations (finding roots)
- Computing $\sqrt{b}$
  - *
  - *
  - *

Success depends on understanding how the computer is getting things done -- and ---

being able to use mathematics.

---

**embedded “performance tasks”**
Conclusions

• Math instructors can (and should) help answer the question
  “After an hour of code: now what?”

• Many meaningful programming assignments can involve mathematics
  • The math programmed is usually just middle school math calculations*
  • There are assignments where
    Math goal “Be able to formulate a plan for the solution”
    =
    CS goal of “Be able to formulate algorithms”
  • Programming assignments can be easily adapted for accessibility

• One has to be careful about creating Math assignments that involve standard programming languages; there are many things to know about programming in order to get a computer program to run.

* This is an expected coincidence. It's what computers are designed to do.
The algorithms discussed are from a branch of mathematics known as “Numerical Analysis”.

The numerical analysis description of the algorithms:

- $\pi$ is approximated by approximating the integral $\int_{-1}^{1} \sqrt{1-x^2}$ using the midpoint method and multiplying by 2.

- The method for solving equations (root finding) is the secant method.

- The method for evaluating square roots is the secant method applied to $x^2 - b = 0$.

- The extra credit square root recurrence is Newton's method applied to $x^2 - b = 0$.

Any introductory text in numerical analysis (and there are many) can provide algorithms that could form the basis of meaningful and accessible programming assignments.
#include <iostream>
#include <cmath>
#include <omp.h>
#include <vector>
using namespace std;

int main()
{
    long N;
    double areaSum;
    double approxPi;
    int threadCount;
    int threadIndex;

    threadCount = omp_get_max_threads();
    cout << "Multi-threaded run using " << threadCount << " threads " << endl;
    cout << endl;
    cout << "Enter the number of rectangles N : " << endl;
    cin >> N;

    vector<double> area(threadCount,0.0);

    long i;
    #pragma omp parallel for private(i,threadIndex) schedule(static,1)
    for(i = 1; i <= N; i++)
    {
        threadIndex = omp_get_thread_num();
        area[threadIndex] += sqrt(1.0 - ((2.0*i - 1.0)/N -1.0)*((2.0*i - 1.0)/N -1.0))*(2.0/N);
    }

    areaSum = 0.0;
    for(i = 0; i < threadCount; i++) {areaSum += area[i];}

    approxPi = 2.0*areaSum;

    cout.precision(15);
    cout << "Results using " << N << " Rectangles " << endl;
    cout << "Approximate value of PI : " << approxPi << endl << endl;
}

Computing $\pi$ : C++ code (multi-threaded with openMP)
Thanks for your attention