COMPUTATIONAL THINKING

A how-to for how-tos

What is a program?

A computer program is a set of instructions

- Specific
- Followed exactly
- Accomplishes a goal

Some programs do big things

- Calculate the orbit of all the planets in a solar system
- Make predictions about economic growth

Some programs do small things

- Start a counter when a specific button is pressed
- Do basic math functions like addition and multiplication

Who designs programs?

Programmers!

What do the coders do?

Who are programmers?

You!

You are programmers!

How do we write programs?

We will write our programs, or code, by following a general approach/process:

- Identify the problem
- Think about the steps to solve the problem
- Brainstorm
    - Outline
    - Code
    - Debug /Optimize
- Repeat
- Present

Identify the problem

This step is easy!

Write down what the issue or goal that you have at the top of your paper

This can look something like this:

- “I don’t have a stopwatch”
- “I need to tell a story”
- “I’m bored”
  - This is a perfectly fine problem!

Think about a solution

Now we will think of solutions we might create, grand or small!

This may look like:

- "I will make a stopwatch that can do everything I want it to! Start, stop, lap, reset, countdown, and even make a noise to tell when time has run out!"
- "I could illustrate my story and have sound elements too!"
- "I guess I could make a game"

All of these are great. Think big, or small! Either way, now that you've got an idea of your problem and your target, were ready to move forward.

Brainstorm

Let’s think of the finer details now.

On your paper, below your dream of your solution, give yourself some space to brainstorm

Here you will decide some of the finer points of your solution:

- You want to be able to press a button to start and stop the stopwatch
- You want the story to be about your cat
- You want the game to be 2D side scroller like Mario
- Anything else that is relevant to your solution that you don’t want to forget as you work

Outline

We’re going to put all of our ideas into a more structured format now, but remember, it is all about what works for you.

We will keep coming back to this outline to refine our ideas.

We will also use it as a checklist.

We will take a much closer look at this structure and process when we work through an example.

Code

Translate your outline into code!

- You don’t have to work in order, you can skip around to complete different parts
- It may not work right away, few programs do. And when they do you still may want to change some things
- Not everything translates easily, you may have to change plans or ask for help

Some tips

There is no shame in the questions game (or google for that matter)

- Everyone thinks differently, you probably have the answer someone else needs and they might have yours!

Coding is ALWAYS a team sport.

- Freely give and freely ask for help whenever possible or needed
- If you really think it should be working but it isn't, take a break or ask someone else to look. NEVER underestimate fresh eyes.

Debug/Optimize

A bug is a flaw in your code.

Debugging is the process of refining your code to make it better, to fix the bugs.

Taking Another Quack At It – rubber duck debugging

- A Coding Duck is a rubber duck that sits on your desk while you work. Explain your code to them and make sure they understand you by putting it in terms you think they'll understand. Tell them why you think it isn't working, talking it out like this may lead you to a solution.
- Remember, your Coding Duck is always there for you, and always on your side. They believe in you and know that you can do it.

Repeat

Repeat by updating your outline as you go, adding more code, and continuing to debug until you are happy with the program

Present

Show off your hard work!

It is important to be able to communicate your program to others. When presenting your program to the world make sure you write up a description that includes the problem you are trying to solve, a little about how your program works, and how it successfully solves that problem.

Always cite your sources if you used other resources and include credit to those who helped you out as well.

Let’s work through a real-world problem

This isn’t exactly a computer program, but the process is still the same

Identify the problem: I am hungry

Dream of a solution: A sandwich to end all sandwiches! And an ice-cold drink! Chips sound good too, maybe even a dessert if I have something sweet on hand.

Brainstorm: alright, well I can only use the items I have on hand so I guess I can’t get too crazy…..but that’s alright.

~~Peanut butter and jelly??? Chicken salad??? My mom used to make egg salad…I thought those were gross~~ oooo I could make a turkey and cheese sandwich and I think I have some of my favorite chips and YAY I checked and I still have some cookies leftover for dessert.

Okay so that’s my plan.

Lunch Program - Outline

For my outline I would bullet point the general steps of what I have to do, leaving a lot of space in between to add in details and revisions.

- Gather ingredients
  - Get bread from pantry, get sliced turkey from fridge, get cheese from fridge, get mayo from fridge, get jalapeño chips from pantry, get cookies from counter
- Prep ingredients
  - Slice cheese, because my favorite kind comes in a block
- Construct sandwich
  - Bread, mayo, turkey, cheese, bread.
  - Cut in half diagonally because it tastes better that way

Lunch Program - Outline

- Add everything to a plate
    - Add sandwich, chips, cookie
    - Don’t forget some tea in a glass
- Eat

Lunch Program - Code

To translate this, we would translate to actions rather than to code. I can make revisions as I go when I run into a bug.

Lunch Program - Bug

- The first bug I hit is that I didn't say which type of cheese. I have cheddar, swiss, parmesan, all sorts. I need to be specific. I need to revise my outline to say grab Havarti cheese from the fridge.
- Next, I find that I didn't get out anything to slice the cheese with. Ill revise to add that I need to get out the grater that I do this with.
- Forgot a butter knife to spread the mayo
- Same with other things. 8 slices of turkey, 6 slices of cheese (they are tiny, it takes 6 to cover 1 layer) a small swipe of mayo.

Every time I find a bug, I will revise my outline

Lunch Program – Outline update

- Gather ingredients
    - Get a plate from the cabinet to the right of the stove
    - Get a butter knife from the drawer to the right of the stove
    - Get the cheese grater with the slicer on it from the cabinet by the fridge
    - Get 2 slices of bread and a small bag of jalapeño chips from pantry
    - Get sliced turkey, Havarti cheese, and the mayo from fridge
    - Get cookies from counter
- Prep ingredients
    - Slice 6 slices of cheese, because my favorite kind comes in a block

Lunch Program – Outline update

- Construct sandwich
  - On the plate, one piece of bread followed by a layer of turkey and a layer of cheese.
  - The top slice of bread gets a small swipe of mayo and then is places mayo side down on the cheese layer.
  - Cut in half diagonally because it tastes better that way.
- Add everything to a plate
  - Add a handful of chips and 1 cookie to plate
  - Don’t forget to get some tea out of the fridge and pour it in a glass

Lunch Program - Present

I was hungry, so I made myself a sandwich. This sandwich has turkey, Havarti cheese, and mayonnaise. It is a simple sandwich, but I know I can add more fun stuff later. I also got chips, cookie, and a glass of unsweet tea. Thank you to my spouse who did the most recent grocery trip and helped me out a lot by making the tea so that I could have it now.

The Real Deal

Identify the problem: I want to make microbit dice

Dream of the solution: I could have the LED lights that show the dice faces. It should "roll" when I shake the microbit!

Brainstorm: I will use a random number generator (RNG) and a conditional statement to show the number. The number will then trigger the LED face to be displayed.

I need to review both of these things, iterations too.

Iterations and Conditionals

Iteration: The repetition of a process.

What do you think these statements will do?

```
Set wheel speed at 0

While speed < 100
        Increase speed by 1
```

Iterations and Conditionals

```
Set wheel speed at 0

While speed < 100
        Increase speed by 1
```

Speed starts at 0, as the program runs it reads each statement in order. The **while** tells the programs to continue looping through until the statement has been satisfied. One time through the speed is now 1. going again now, 1 is less than 100 so now increase to 2. this continues until the while loop is "broken" meaning that the statement is no longer true. Speed is now equal to or greater than 100.

Iterations and Conditionals

Conditional: dependent on something else, certain conditions apply.

```
Run RNG and set # = Dice

If Dice = 1
        then show Dice Face 1

If Dice = 2
        then show Dice Face 2
……
```

Iterations and Conditionals

```
Run RNG and set # = Dice


If Dice = 1
        then show Dice Face 1
```

This statement says that IF something is true (meets the condition) THEN perform this action. This also means that if the condition is not true, the program will NOT perform that action