Candy dispenser design challenge

Brainstorm and planning notes for

____

name

Taking on the design challenge

One of the best things about learning computer science is that it isn't just about writing programs. You can use computer science to create all sorts of things, including different inventions using your Edison robots.

This design challenge is to create a candy dispenser using one or more Edison robots.

What will you create? How can you build it? What will the computer program you need to run your creation look like?

Inventing something with Edison and one of Edison's programming languages is a big task, but you can make it a lot more manageable by **decomposing** it into smaller pieces and doing each thing one at a time.

Jargon buster

In computer science, **decomposition** is the process of breaking a complex problem or system into parts that are easier to understand, manage and solve. Breaking a large problem into smaller, more doable tasks makes it easier to determine exactly what needs to happen and in what order each task needs to be done.

Breaking any major job up into smaller parts is one of the best ways of approaching it. If you get stuck or feel overwhelmed when tackling a project, take a minute to think about how you can use decomposition to break it down into smaller tasks.

When it comes to creating your own inventions with Edison, you can also use **iterative testing** through something called the **design-build-test cycle** to help you.

Jargon buster

The **design-build-test cycle**, which is also sometimes called the design-build-test-learn cycle, is a process where you design something, build it and then test it out to see what works and what needs to be improved. You then take what you learn from the test and apply that back to the design. Because it is a cycle, you keep repeating each step, applying what you learn to make the next cycle better, until you are happy with the outcome.

Making and testing many versions of something, applying what you learn and making changes each time, is known as **iterating** or **iterative testing** and is a common practice when developing new things in computer science. Technology companies use the design-build-test cycle and iterative testing all the time when they are coming up with new products.

It's pretty unlikely that your first design will work perfectly. That's okay! By using iterative testing and working through the design-build-test cycle, you can keep refining your idea, applying what you learn and making improvements.

Remember that a major part of computer science is using **computational thinking** and a major part of computational thinking is problem-solving!

Jargon buster

Computational thinking means thinking about a problem or task in a similar way to how a computer thinks. It is a way of logically working through problems, decomposing them into smaller pieces, finding patterns, and then using the information to come up with a step-by-step solution.

In other words, computational thinking is a way of planning, problem-solving and analysing information the same way a computer does.

Another big part of computer science is being creative and trying new things. You never know what you might be able to create until you try!

Task 1: Brainstorm

Before you can get started designing, building and testing something, you need to come up with some ideas. Coming up with ideas is all about being imaginative, using your creativity and thinking about possibilities.

Brainstorming is one way to kick-start your thinking to generate ideas. When you brainstorm, you are trying to get thoughts to flow freely. All ideas are acceptable, no matter what they are. Don't make decisions about whether the ideas are possible or judge the ideas while you are brainstorming.

Let's try brainstorming some Edison candy dispenser inventions. Using the activity sheet on the next page, you need to come up with six different ideas of something you could create and program using Edison robots. You will only have 45 seconds for each idea before you need to move on to your next idea.

Capture your ideas however you like. You can draw something, write down the idea in words, or do a bit of both! You are not allowed to NOT come up with ideas, however. Remember, there are no 'bad' ideas during a brainstorming session!

Set a timer for 45 seconds and get started on your first idea. As soon as the timer goes off, reset it and move on to the next idea until you have all six ideas done.

After you finish brainstorming, analyse the ideas you came up with. Looking at all your ideas together, you may decide that some of your ideas are better than others. You can then choose one of these ideas to use and move on to designing your creation.

Activity sheet: Six ideas

1

2

3

4

5

6

Task 2: Design

Once you have at least one idea you think might work, you can start planning and designing. You will need to decompose your idea into smaller parts, design each part, and plan how to tackle each one. At a minimum, you should break your idea into two parts: the physical design using Edison and the program design.

There are lots of different tools you can use to help you design and plan. You can draw out sketches or diagrams, you can create a storyboard, or you can write up an overview. When you design your program, you will probably also want to use **pseudocode** to help you plan how the program will function. You can always use a mix of approaches too – whatever works best for you!

Jargon buster

Pseudocode is a way of writing out a program in a simple, easy-to-read format. Instead of worrying about syntax, pseudocode uses normal words to describe what the program will do.

Pseudocode looks a bit like a simplified programming language, but it isn't based on any specific programming language. That's why pseudocode can be used to plan programs in any coding language.

Using one of your ideas, work out a design for both the physical creation and the program.

Here are some of the things you should think about when designing:

- What will your creation do?
- What will your creation look like?
- What materials could you use to build your creation? How will you attach these to Edison?
- How will the program control Edison to make your creation work?

Remember, this design is just your first one. There's still a lot of learning to do! Your design might not be able to predict everything, and you might have some things you aren't sure about. If you have any questions about how parts of your creation or program can work, make a special note of these questions. These are the types of things you will want to pay close attention to when you build and test.

1. Give your project a name and write a brief description of your idea.

____

____

____

____

2. What are the different parts you have decomposed your project into to help you design it? Remember, you should have at least two parts: the physical creation using Edison and the program design.

3. Work out the design of your physical creation. Use extra space if you need.

4. Design your program (in other words, your code) for your creation. Use extra space if you need.

Extra notes: What have you not been able to work out in your design? What questions do you need to pay extra attention to when you build and test? Note them in this space.

Now, it’s time to try!

Is your idea possible? Can it be built? Can it be programmed? The only way to know is to try!

Try turning your idea into a reality by using your design as a guide to help you build and program your creation. Test your invention to see what works and what doesn’t. Take what you learn from your test and apply it back into your design. Keep repeating the cycle, applying what you learn to make improvements to each iteration.