User Manual

Niall Kelly (20461772) & Adam Gray (20364103)

Project Title: START Web Application

Supervisor: David Sinclair

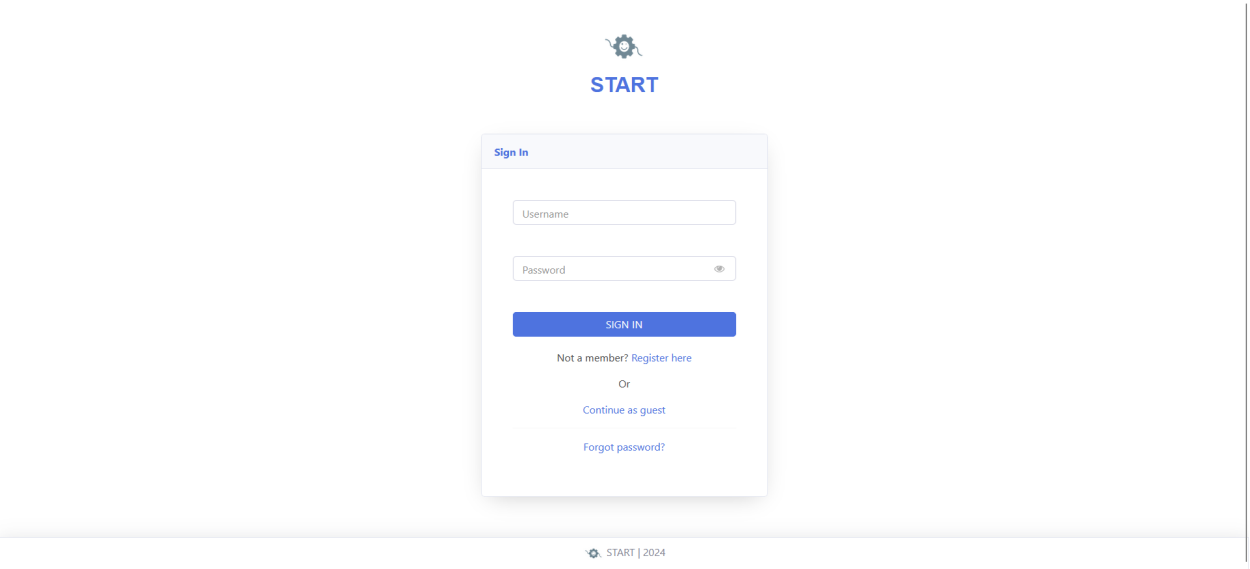
Introduction

START is a programming language specifically designed for beginner programmers, and aims to teach them programming concepts with the use of simple syntax and minimal but strong features. The START Web App contains everything a beginner would need to start their programming journey, from a purpose built IDE designed for START, to learning resources and problems to boost your ability, everything a beginner needs to become a great problem solver through START can be found here.

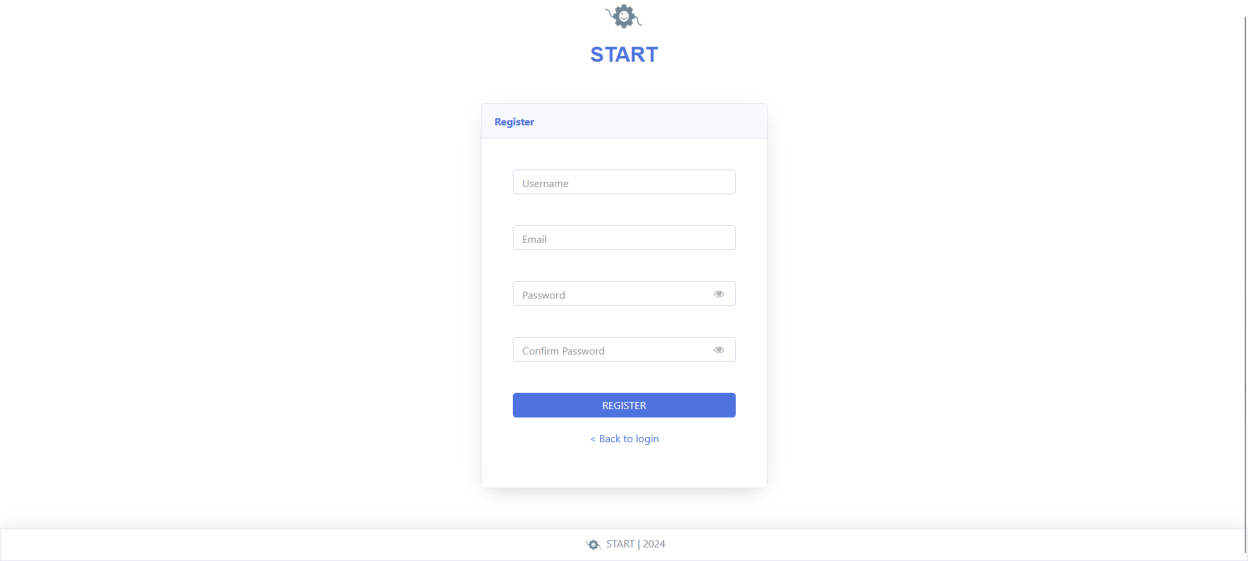
Navigating the Web App

Login or Register Pages

When first visiting the web app, you will be greeted by our login page. If you wish to make an account, you can click “Register here” to create an account. If you already have an account, simply enter your details and click “SIGN IN”. If you do not wish to make an account, but would like to use the app, please click “Continue as guest”.



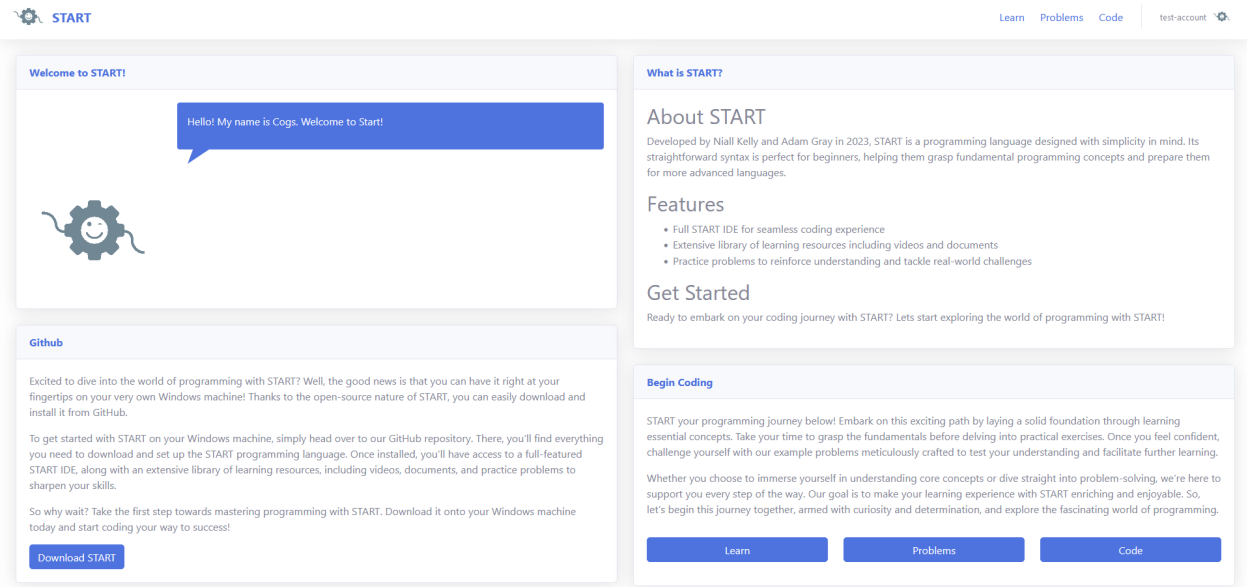
If you have chosen to register you will be seeing the page shown below. To register simply fill in your details and click “REGISTER”.



Home Page

As seen below is our Home Page. We made the decision to keep this page minimal, as we did not want to overwhelm you when you first visit the web app, and want to keep your focus on learning to code as quickly as possible. From here

you can get some important information about START and are guided to start your programming journey from one of the links on the page.



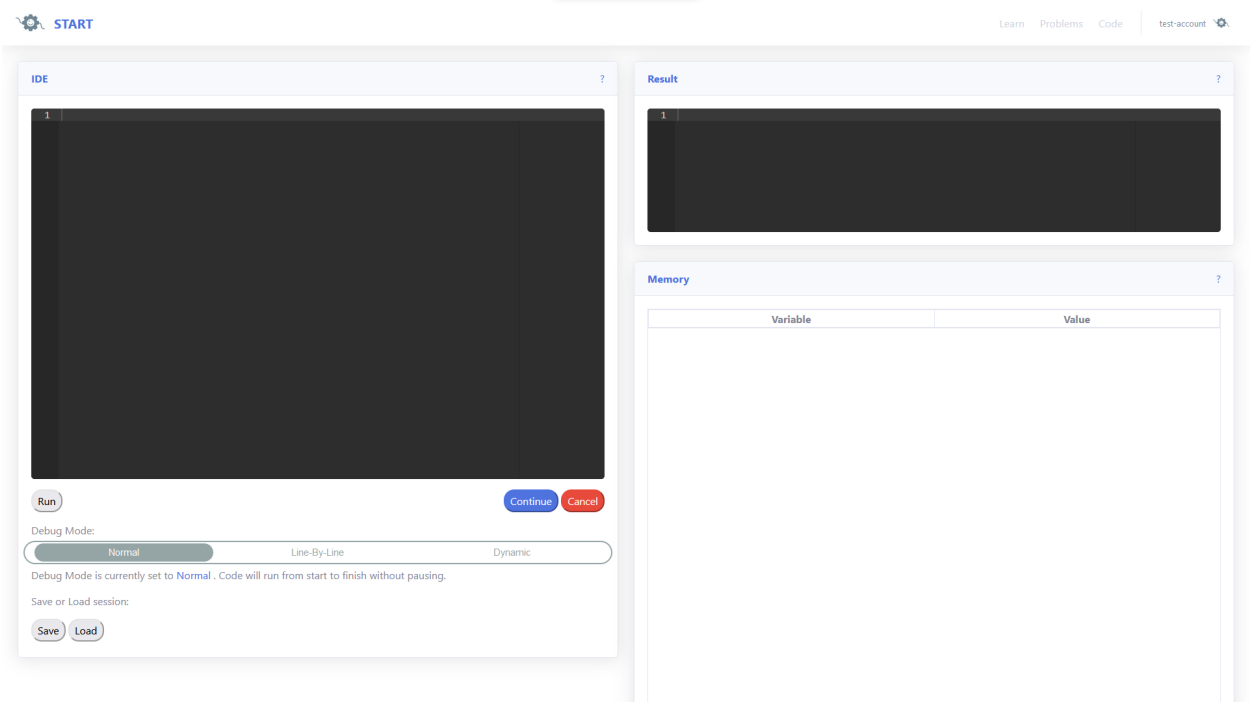
IDE Page

Once you are ready to write your first program you will see this page below. On the left of the screen, the large text editor is where you will write your START code. When you write code, you will see that some of the text will be highlighted, this syntax highlighted better helps you to visualise the parts of your code.

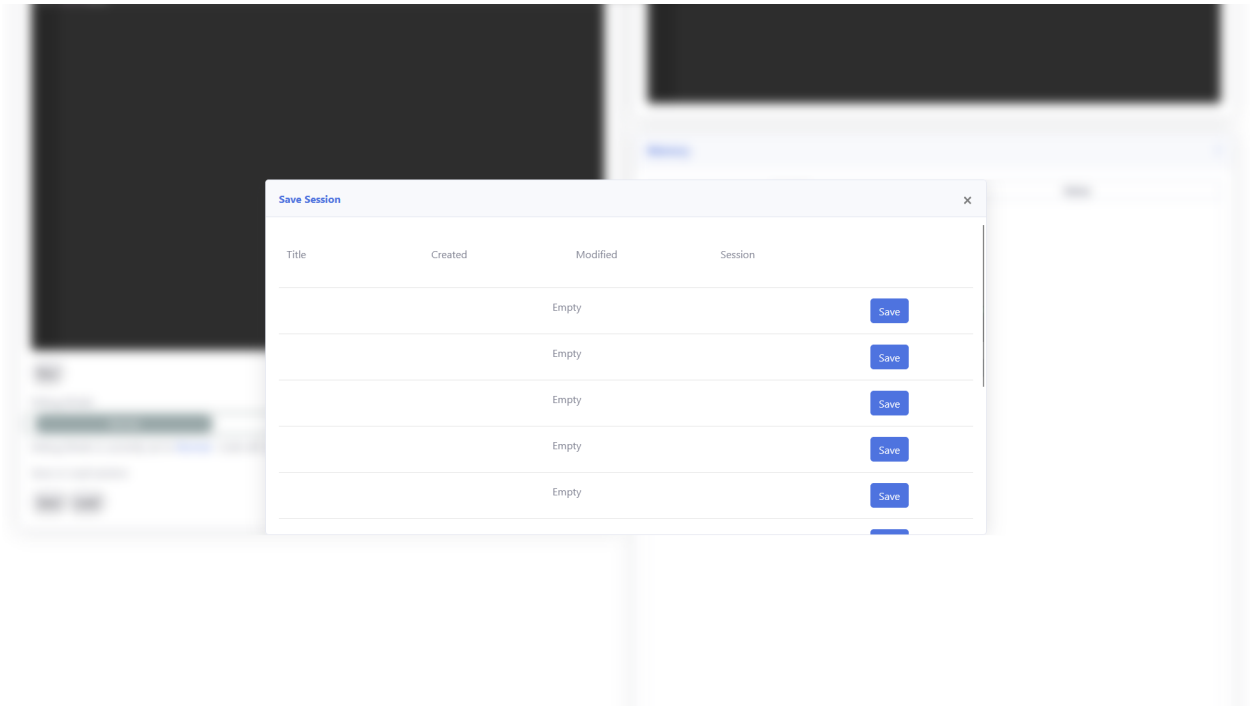
Below the text editor you have 3 main buttons, “Run”, “Continue” and “Cancel”. Run is used to run your code. Continue and cancel will be discussed later in the “Using Debug Modes” section of the manual.

Below these buttons is the mode selector, which again will be discussed later in the “Using Debug Modes” section of the manual. Here we also have the “Save” and “Load” buttons, which allow use to save our current code, or load code we have previously written.

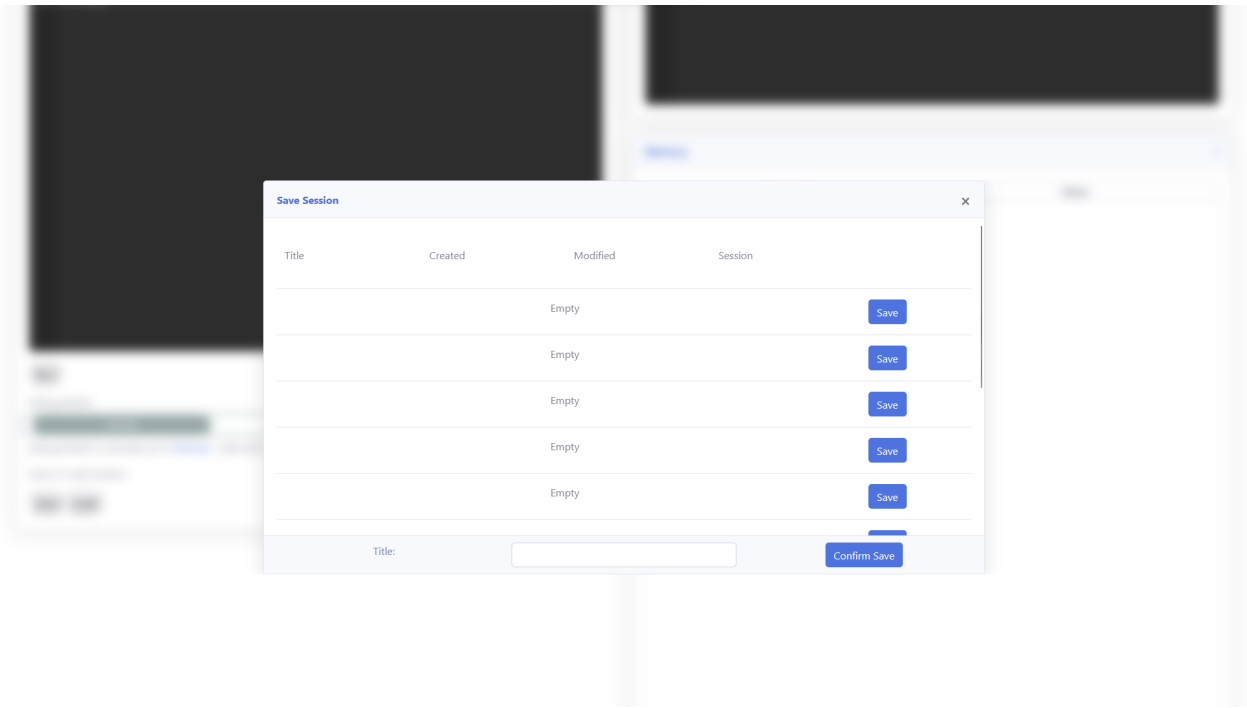
To the right of the text editor at the top of the page, is the result area. This is where the result of your program will be outputted to. Below this is the memory map, which will show you the value of each variable in your code.



Lets say a user has written a program they wish to save, and they click “Save”, they will see the following page:

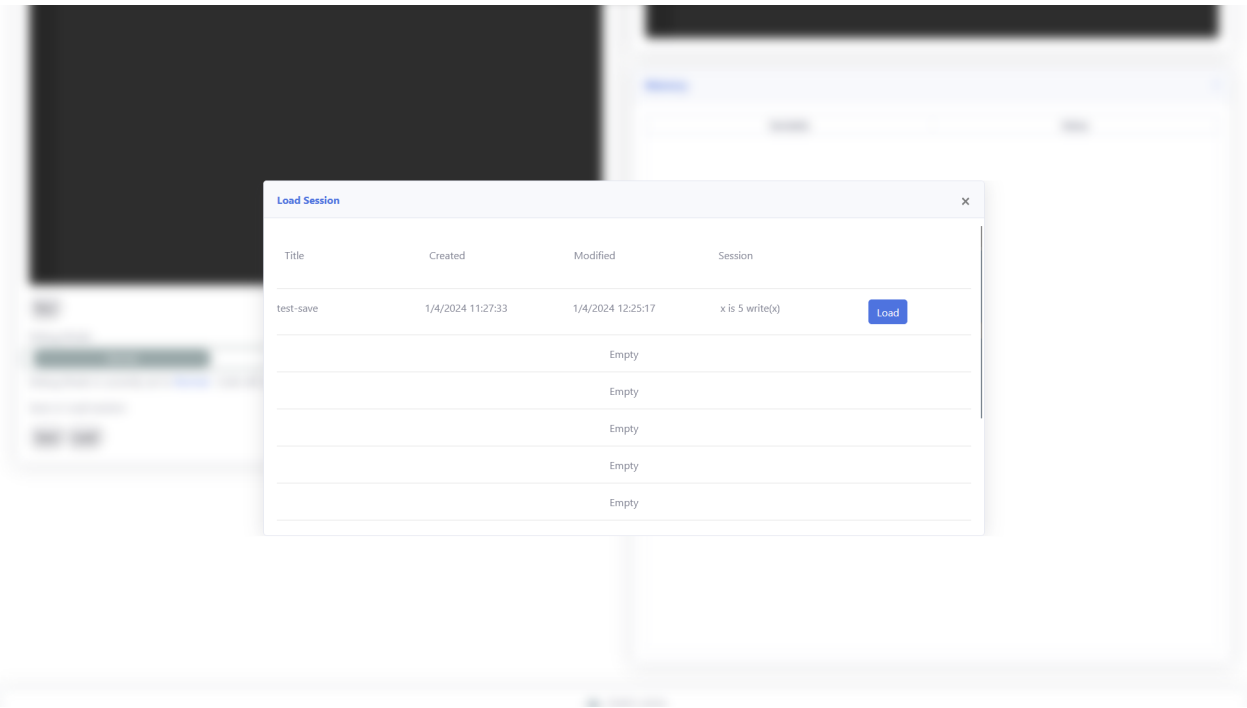


The user can then select a session spot to save in, lets say they chose the first in this example, they will then see the following updated page:



The user can then enter the title of their save, and can then click “Confirm Save” to save the code, and will be returned to the IDE Page.

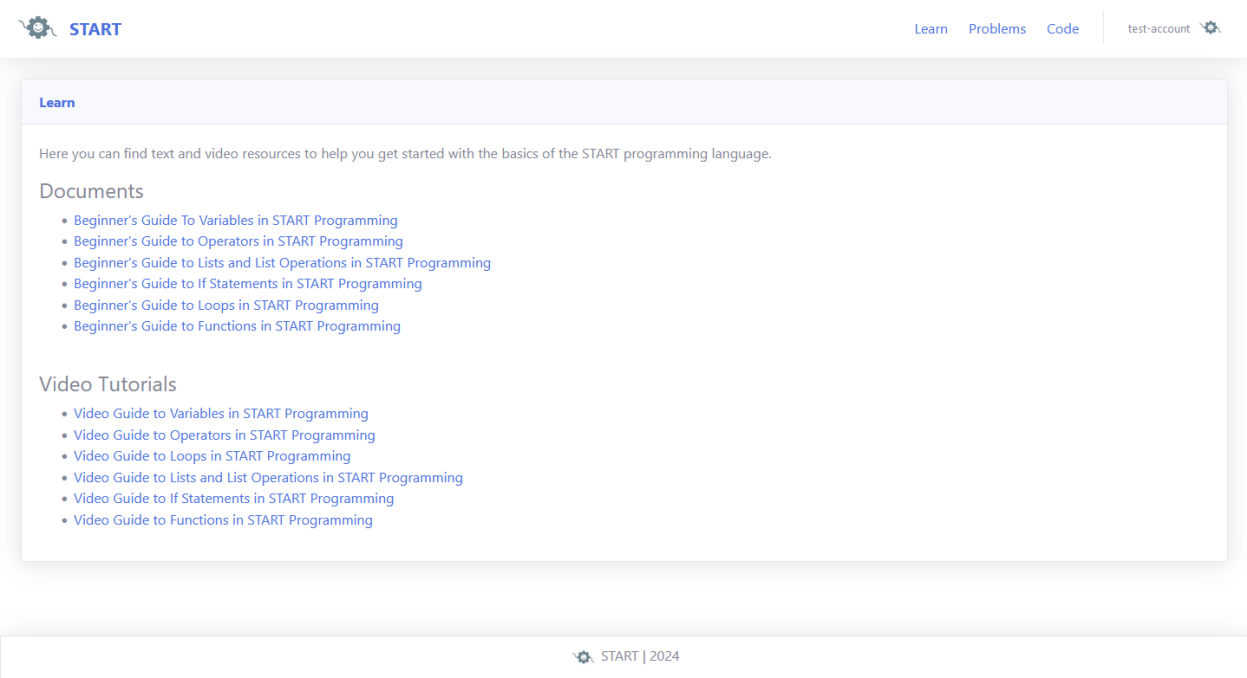
If the user returned to the web app the following day, and wished to reload the previous save. After clicking the “Load” button, they will see their previous saves, as seen below.



After clicking “Load” on the session they wish to load into the text editor, they will then see the saved code in the text editor.

Resource Pages

The page seen when clicking “Learn” is a simple one, containing links to either documents or videos, this was done purposely to keep the you focused on the problem you are trying to solve or learn about.



From here, if you click on a learning document, you will see a page similar to the following one, containing information about the chosen topic:

START Learn Problems Code test-account

Learn

Beginner's Guide to Variables in START Programming

Welcome to the exciting world of programming! One of the fundamental concepts you'll encounter as you start your journey into the world of coding is variables. Variables are like containers that hold different types of information or data in a program. In this guide, we'll walk you through what variables are, why they're important, and how you can use them in your programs.

What is a Variable?

Think of a variable as a labeled box where you can store information. This information can be numbers, text, true/false values, or even more complex data structures like lists of information.

Types of Variables

In programming, variables can hold different types of data. Some common types include:

- **Integer**: Whole numbers, like 1, 10, -5, etc.
- **Float (Floating Point Number)**: Numbers with a decimal point, like 3.14, -0.5, 10.0, etc.
- **String**: A sequence of characters, enclosed in quotes, like "hello", "world", "123", etc.
- **Boolean**: Represents either true or false.
- **List**: A collection of items, which can be of different types, enclosed in square brackets and separated by commas, like [1, 2, "hello", True]. Lists can contain data of all the same type, or a mix of data types.

Declaring a Variable

To use a variable in your program, you first need to declare it. Declaring a variable means telling the computer what type of data the variable will hold and giving it a name.

In many programming languages, you declare a variable by specifying its type followed by its name and then assigning a value to it using the `=` sign. However in start we have simplified this for more readable code, using the keyword `is` to assign a value. Some example variables in START can be seen below:

```
age is 25
piValue is 3.14
message is "Hello, World!"
isRaining is true
myNumbers is [1, 2, 3, 4, 5]
```

In the examples above:

- `age` is an integer variable holding the value 25.
- `piValue` is a float variable holding the value 3.14.
- `message` is a string variable holding the text "Hello, World!".
- `isRaining` is a boolean variable holding the value true.
- `myNumbers` is a list variable holding a sequence of numbers.

Naming Conventions for Variables

When naming variables, it's important to follow certain conventions to make your code readable and understandable:

- Start with a letter or underscore (_), followed by letters, digits, or underscores.
- Never use special characters like $, @, #, etc.
- Use meaningful names that describe the purpose of the variable.
- Use camelCase or snake_case for multi-word variable names.

```
goodVariableName is 10
another_variable_name is "Your name here"
```

Using the `write()` Function

To output the values of your variables, you can use START's built-in `write()` function. This function outputs the contents of a variable (or a string given to the function) to the console for you to see, for example:

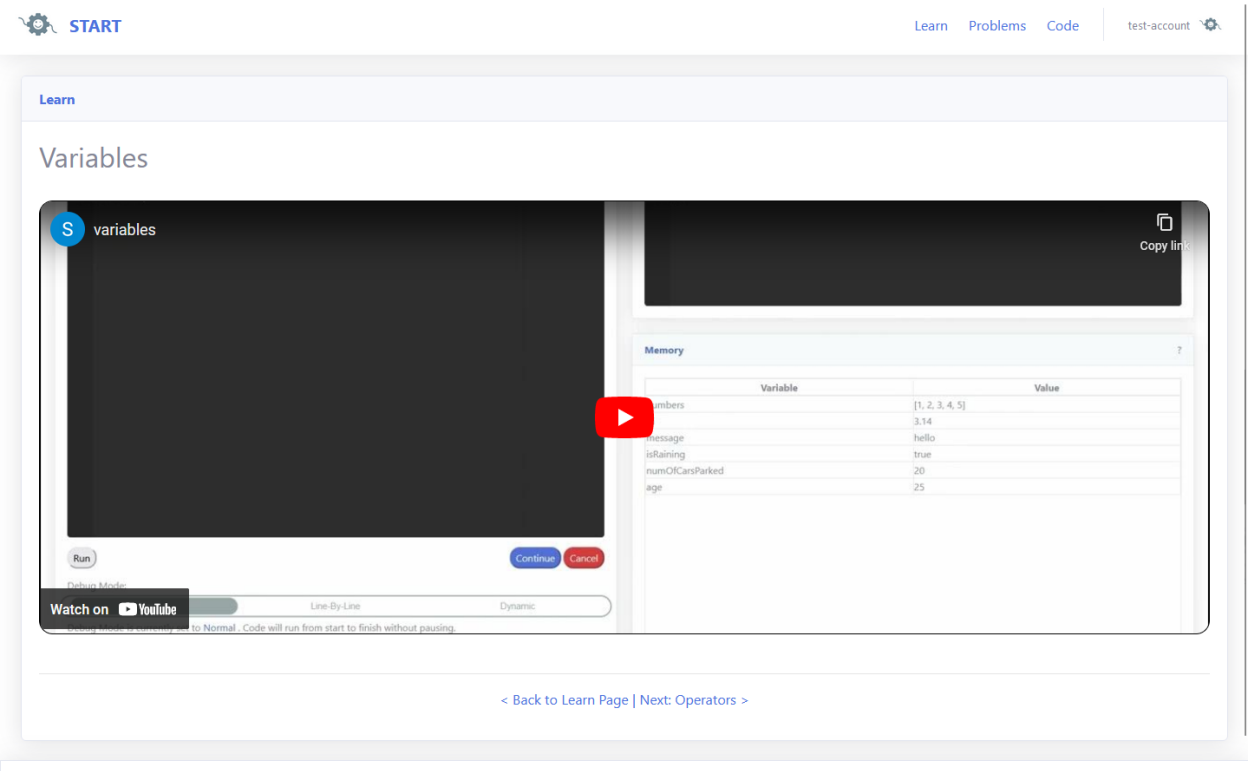
```
write(content)
```

`content` is the information you want to output to the screen. This can be a variable, a string, or any other valid expression.

Example Usage

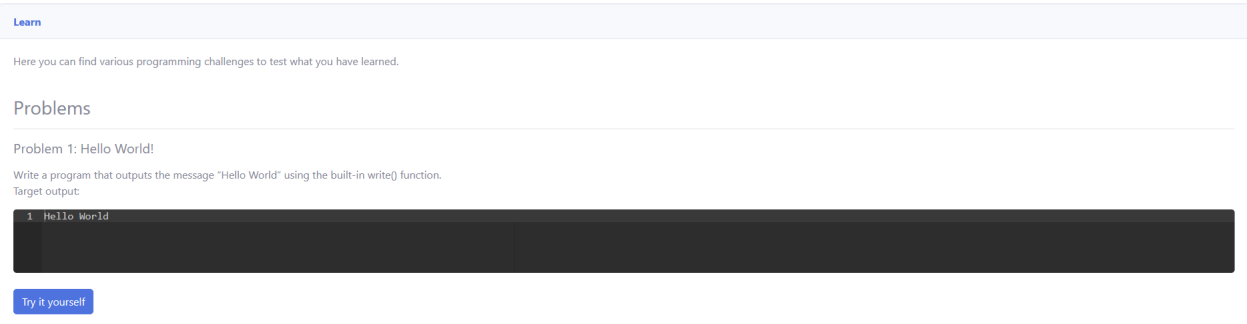
```
name is "John"
write(name)
string is " is"
```

Finally the video links take you to a page containing an embedded video for you to watch, as seen below:

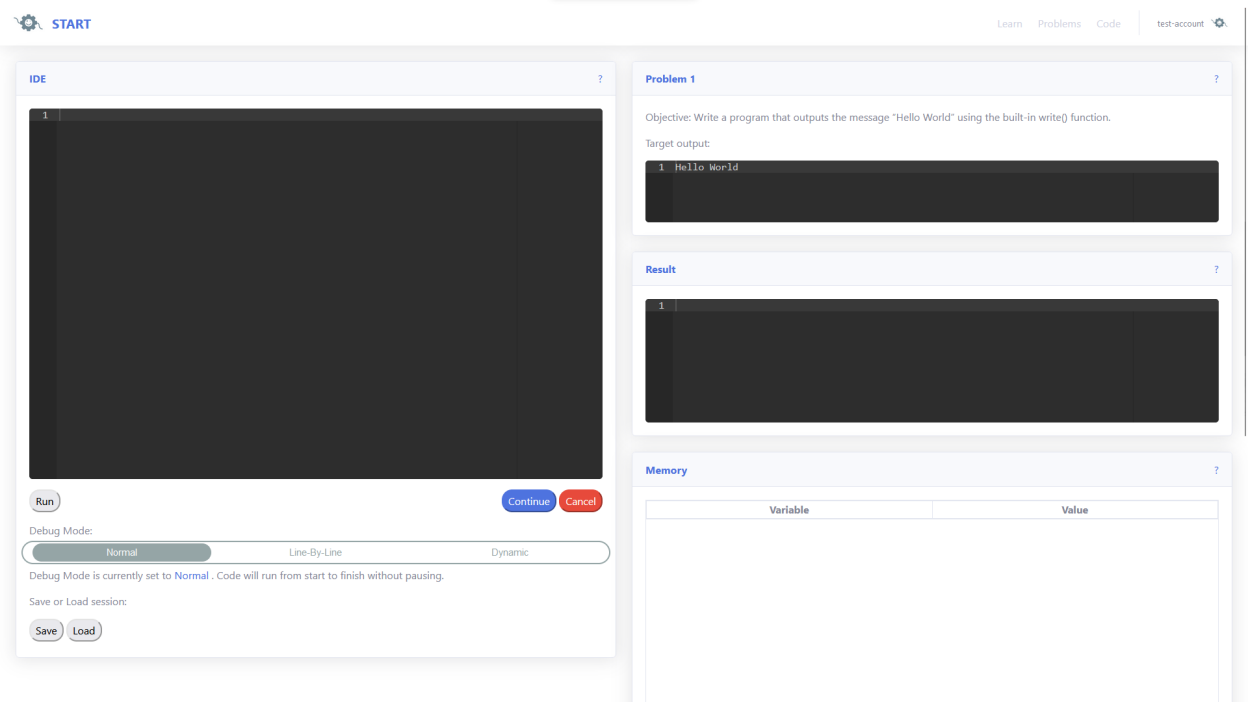


Problems Page

The problems page contains several problems you can attempt to solve when you feel up to the challenge. Looking at the first one as an example, we see the title of the problem, followed by a description of what the problem is asking us to do, as well as a target output we want to achieve with our program.

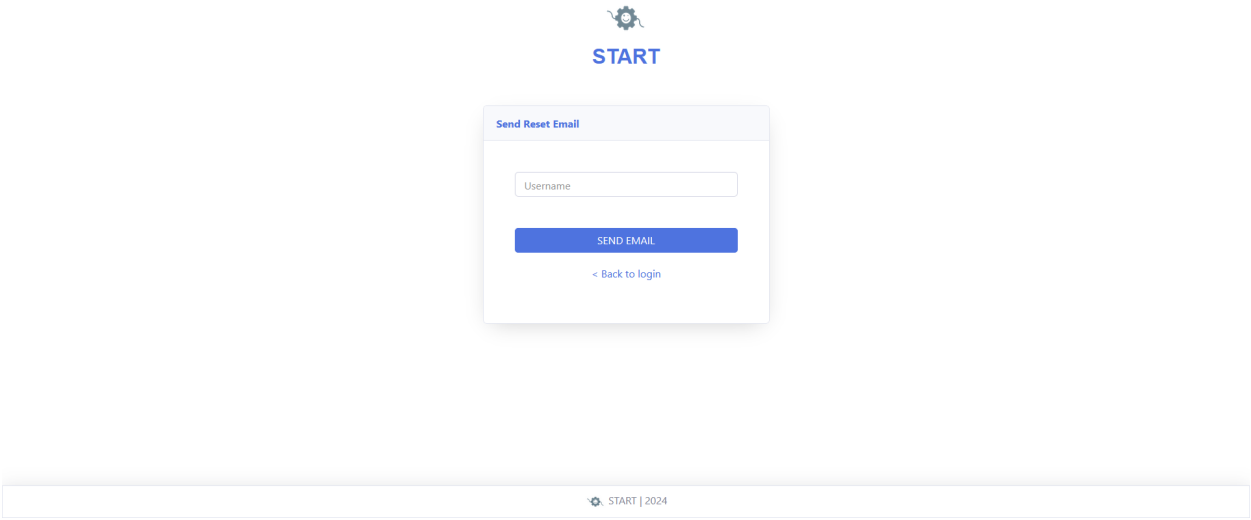


Upon clicking “Try it yourself” you will be greeted with the following page. It is similar to the IDE page, however it contains a target output to achieve above the result box. The idea here is to try to match your output to the output shown here.

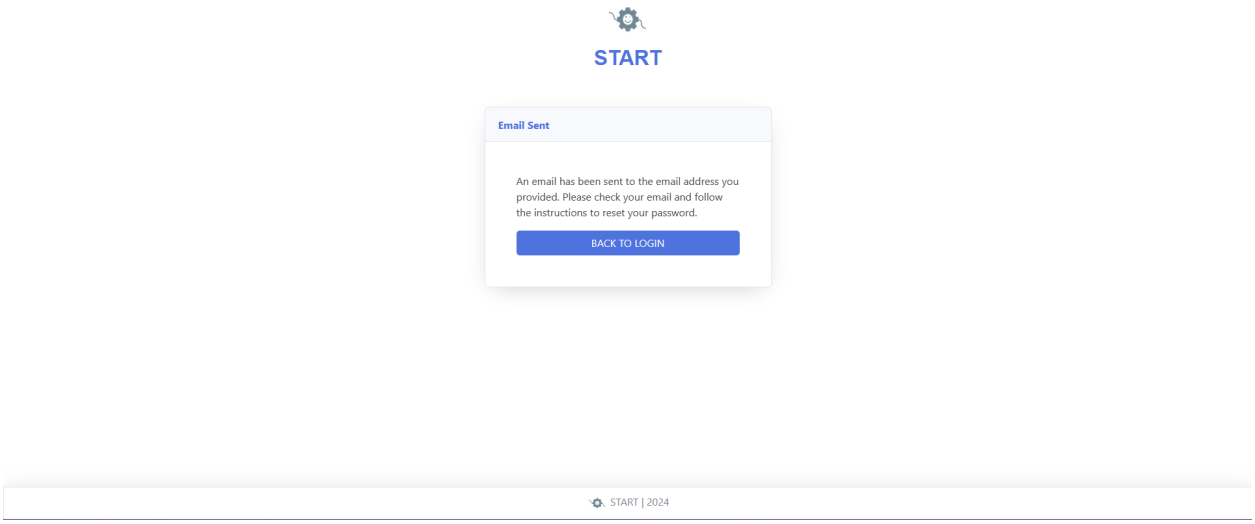


Reset Password Pages

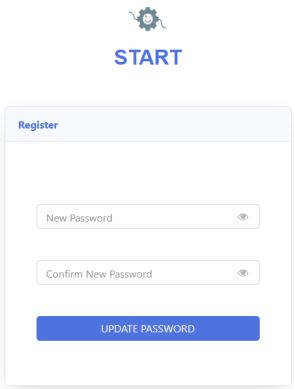
After clicking “Forgot password?” on the login page, you will see the following page:



Upon entering your username for the account you want to reset the password for, and click “SEND EMAIL” you will then see the page below.

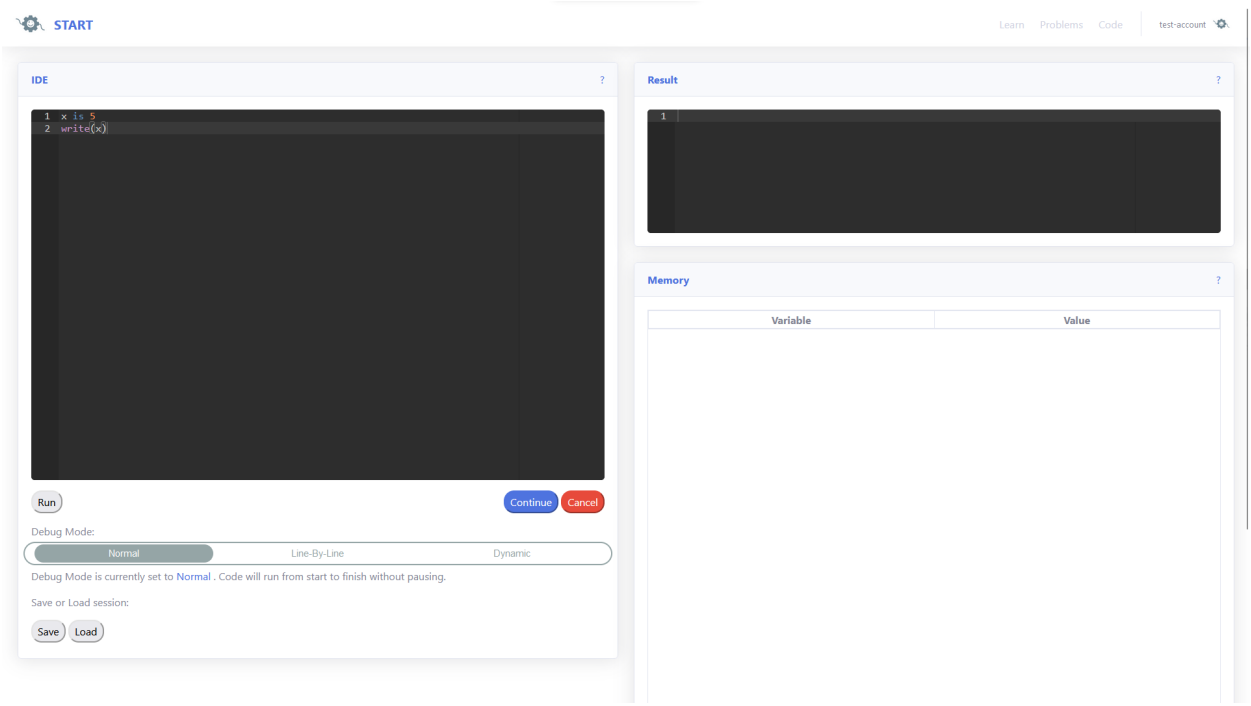


Once you have gone to your email provider and clicked the linked sent to your email, you will see the below page. From here simply enter and confirm your new password and click “UPDATE PASSWORD”. Once done you will be sent back to the login page and can then login as normal.

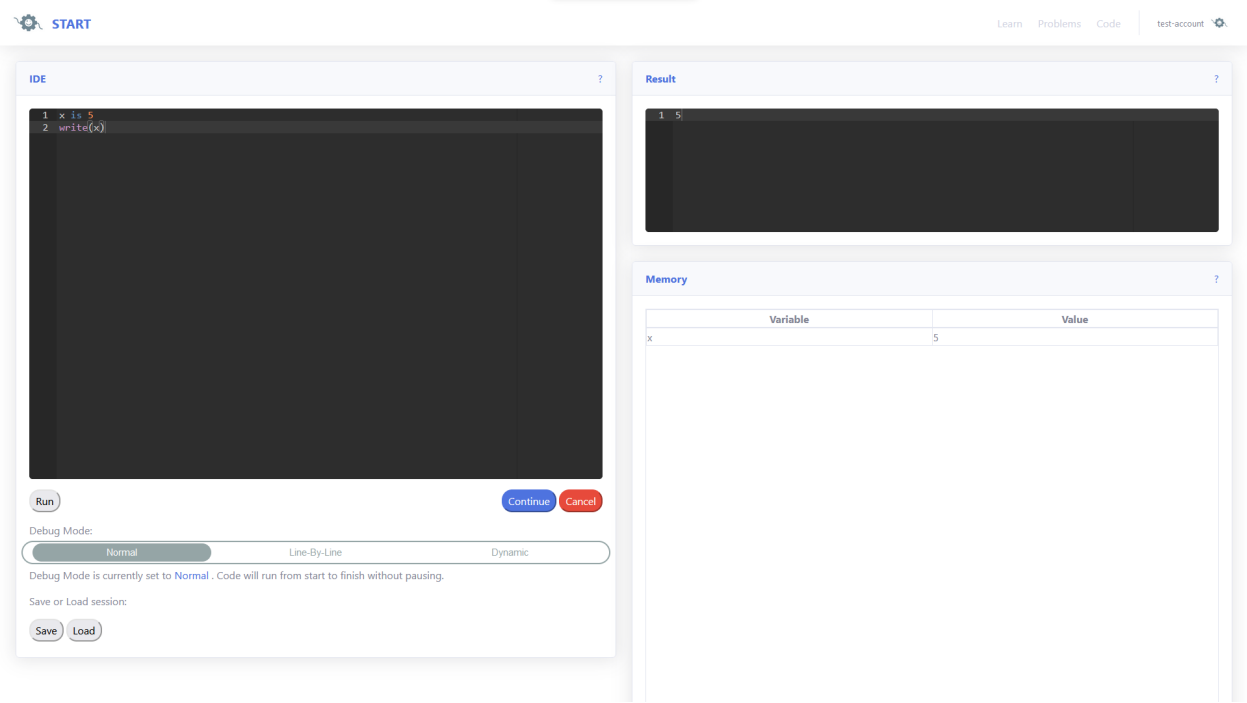


Running Your Programs

Looking at the below image, and seeing we have this simple program we want to run.



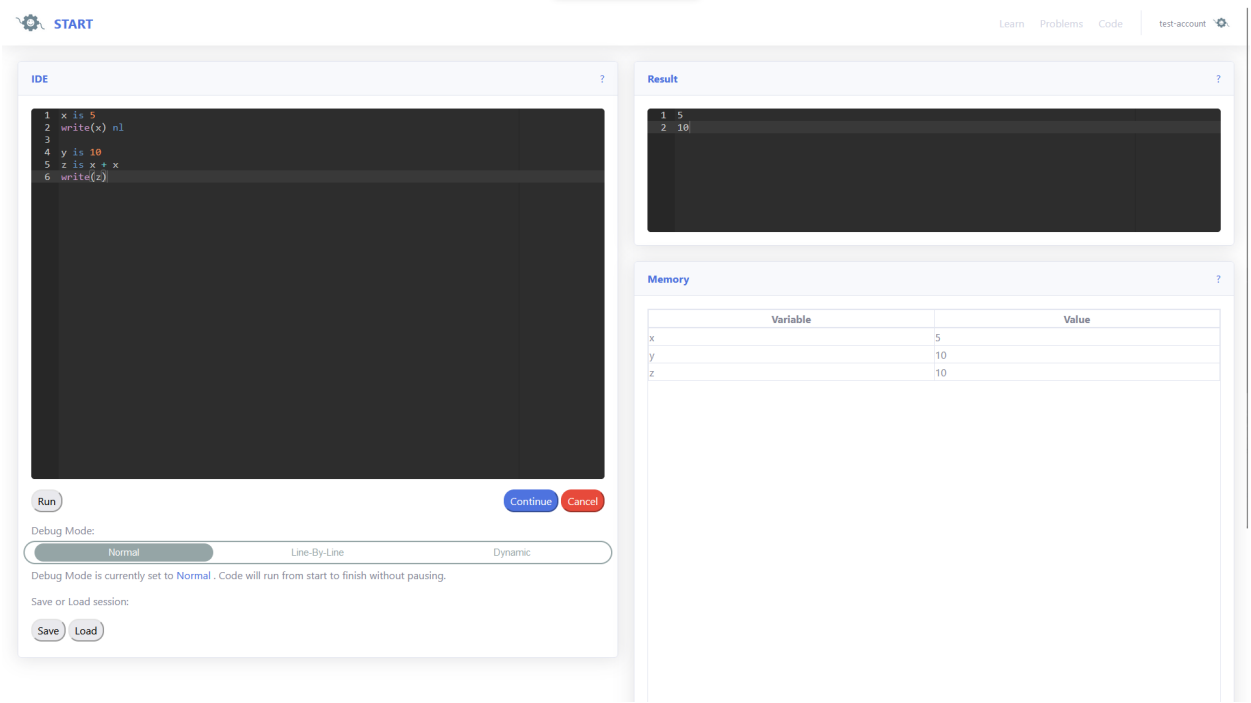
Once we click the run button, the page updates to be the following:



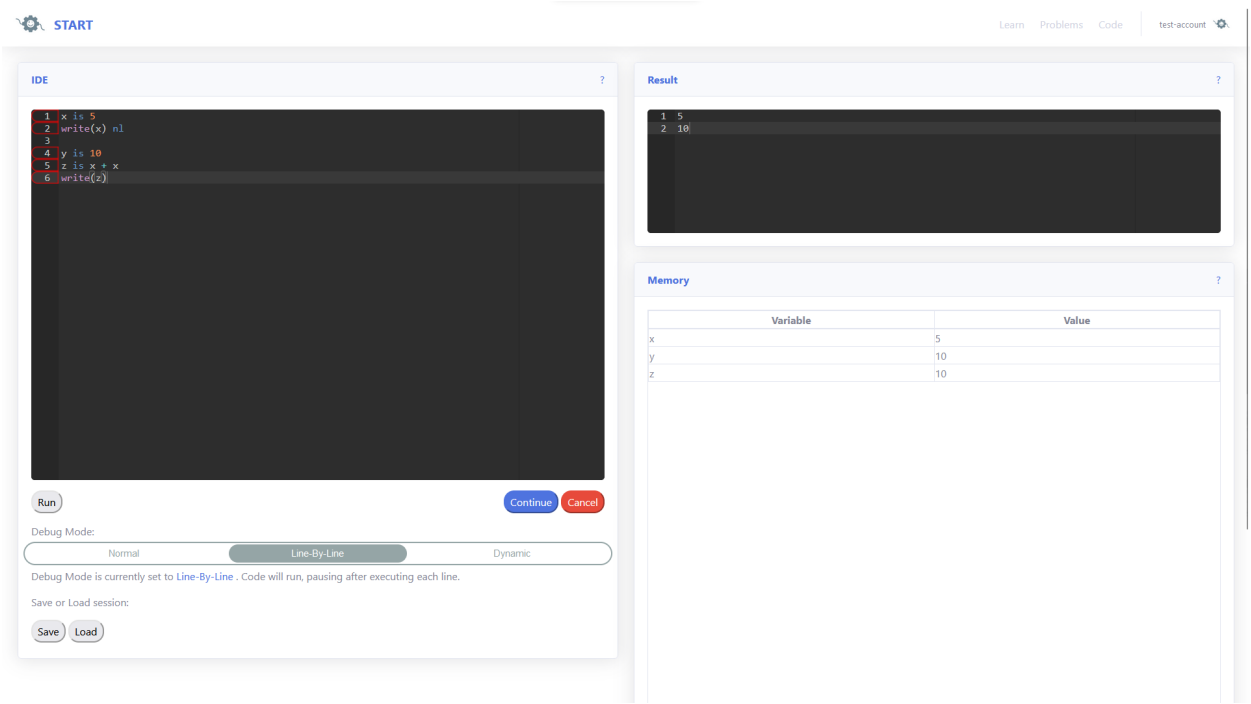
As we can see the value of `x` was outputted to the Result box, as well as the Memory Map being updated with both the variable name and value.

Using Debug Modes

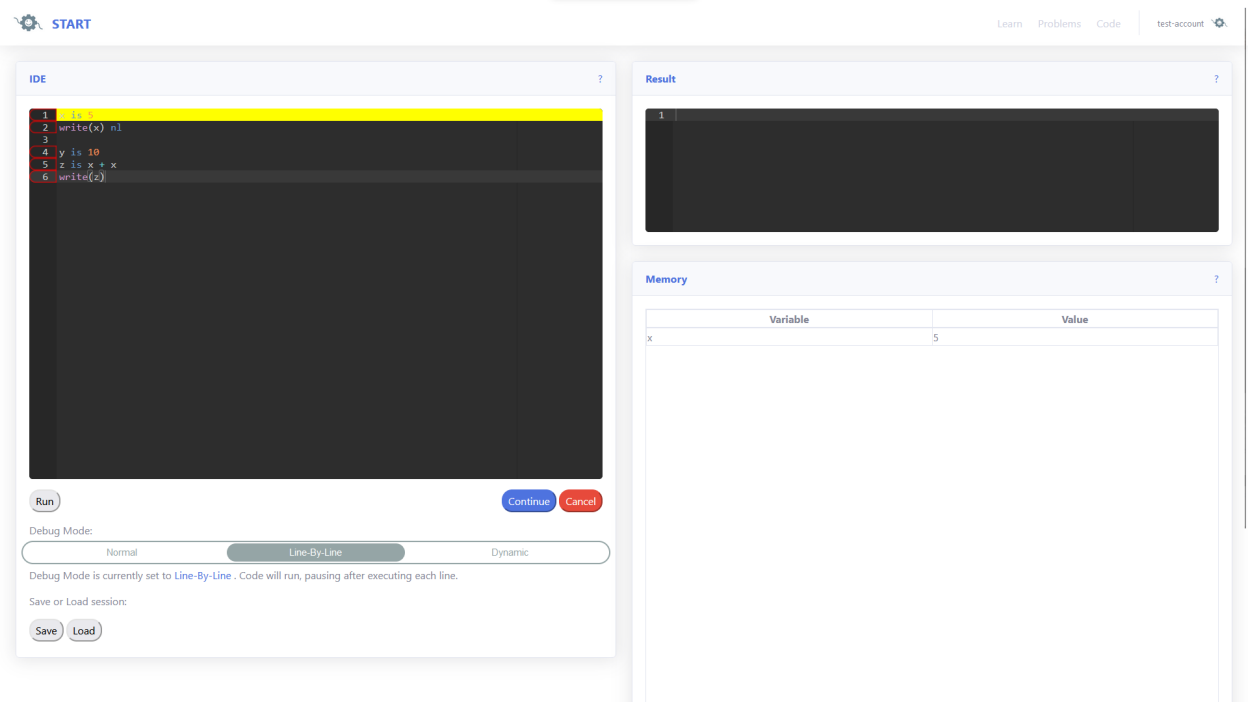
Looking at the following problem written in the text editor, and we cannot figure out why `z` has value 10 and not 15.



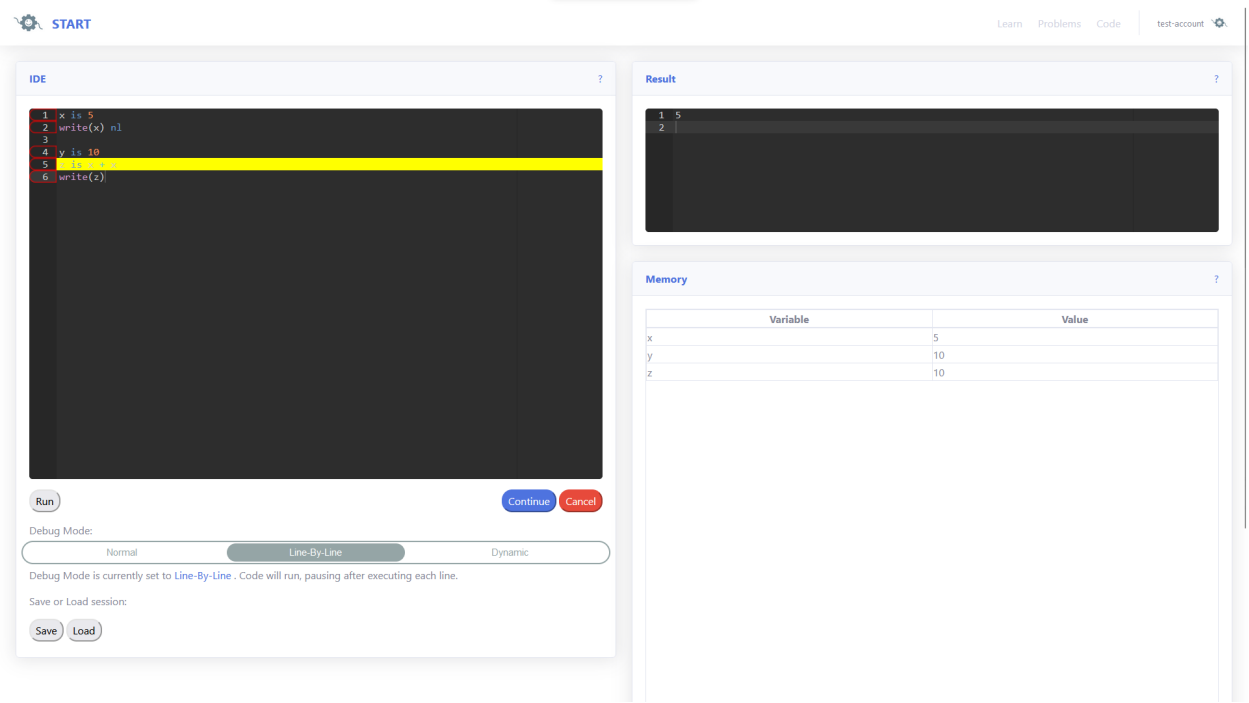
We might think it best to use the “Line-By-Line” mode to slowly see the program execute. By clicking the “Line-By-Line” button the page looks like the following:



As we can see the line numbers with code on them become highlighted in red. This indicates that these lines will be stopped on. Upon clicking run, we see the following:



As we see, Line 1 becomes highlighted to indicate this is where the program currently is, and we see that the Memory Map has been updated as Line 1 has been executed. By clicking “Continue” until we get to line 5 we see the following:



Lets now say we have realised that we have added `x` together twice, instead of adding `x` and `y` to get value `z`, and we want to stop execution as we have found the bug in our code. We can now click the “Cancel” button, and the code has now stopped executing.

“Dynamic” mode also works similarly, however we as users get to chose the lines that the breakpoints are present on, which allows for faster debugging. For beginners it is recommended to use “Line-By-Line” until they feel more confident in their debugging skills.