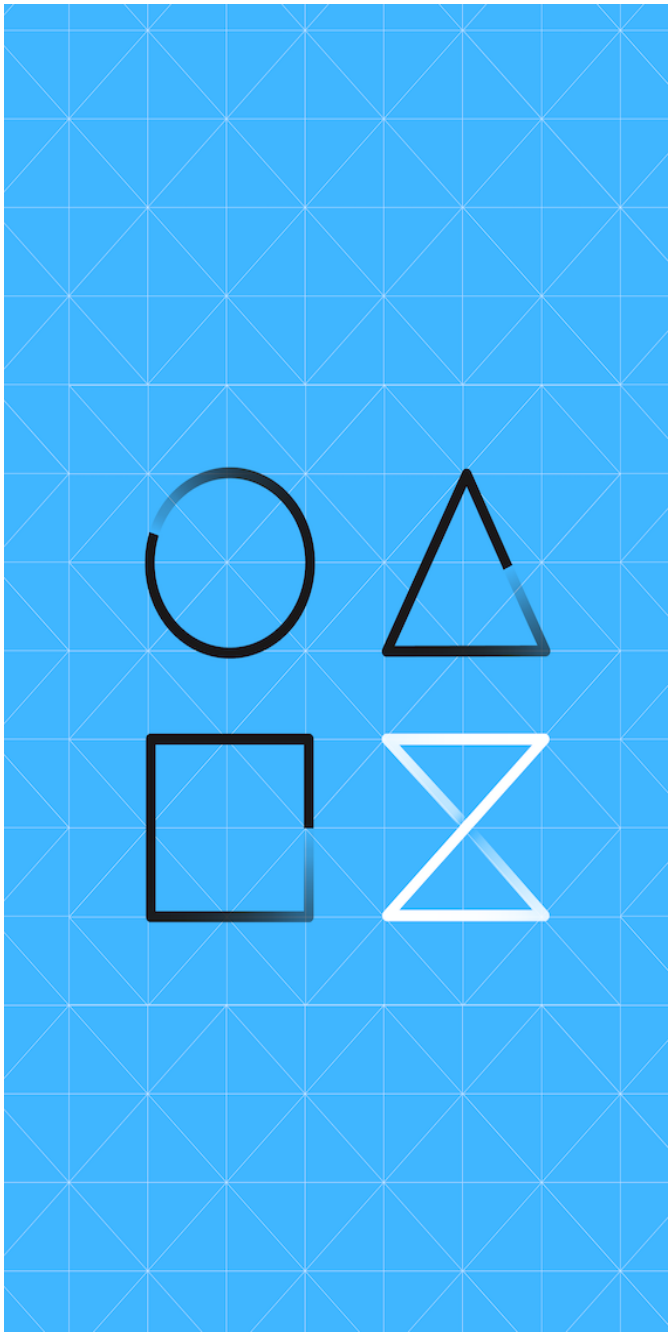


Fundamentals *241112-1738*

REXX1

Enterprise IT Duct Tape

241112-1738

ENTERPRISE IT DUCT TAPE

The Challenge

In this challenge, you will meet Rexx ("Restructured Extended Executor"), a programming language known for its simplicity, power, and relative ease of use. You will dive into how to run a Rexx program from a command line, as well as how to start up a TSO 'Address Space' to run an interactive Rexx program.

To interact with TSO, you will need the ZOWE Command Line Interface (CLI) installed.

Before You Begin

This challenge requires some configuration to your terminal environment.

Make sure you do the ZOWE CLI installation steps.

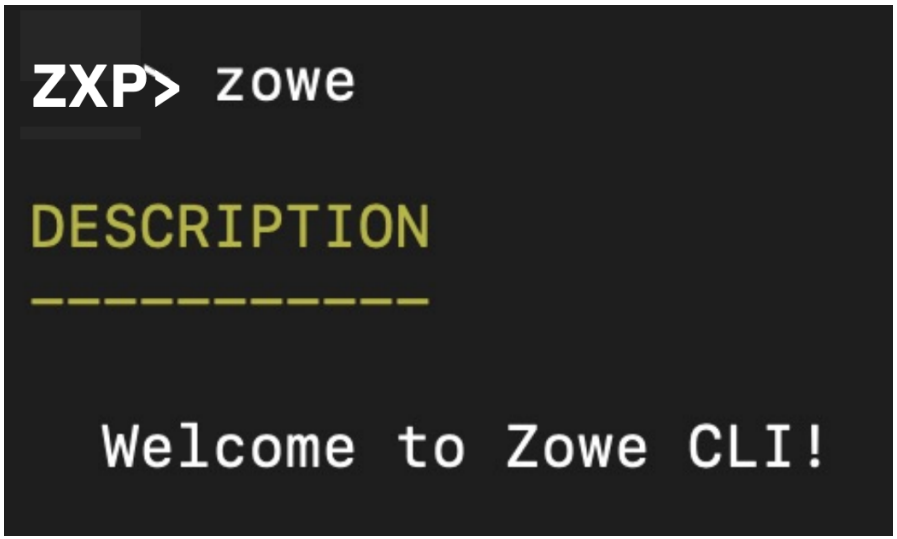
Investment

Steps	Duration
12	60 minutes

1 INSTALL THE ZOWE CLI

So far, you have been using the Zowe Explorer plugin for VSCode, but Zowe can do much, much more, and is responsible for bringing so much more to the mainframe.

To be clear, you will be installing the Zowe Command Line Interface (CLI) on your own computer, *not* on the mainframe.



You will use the Zowe CLI to work with the z/OSMF service which is running on the mainframe, but you'll be driving most of this challenge from your own computer.

Linux users may need to do a bit of exploring to find what works on your specific system, but it should look closer to the Mac steps, just substituting your correct shell profile file.

Windows users may need to modify the user PATH environment variable to be able to successfully run zowe as a command - more on that later.

2 ZOWE NPM INSTALL ON MACOS

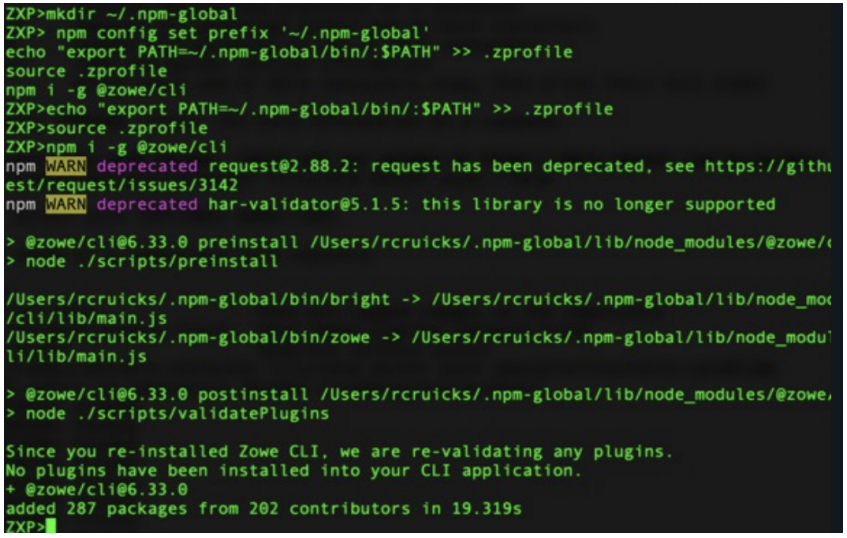
READ this whole section before installing the ZOWE CLI

First, exit from your VSCode application - make sure VSCode is not running.

Because the Zowe CLI uses node packages, it requires that a supported version of Node.js is installed and properly accessible.

In order to use node packages in the operating system, you will need to load them into a '.npm' global directory which can be accessed by regular users.

The following steps will set that up, tell npm (the Node Package Manager) to use it, and include that in the normal list of places the operating system looks for programs to run.



For most users of MacOS, these should do the trick.

1. `mkdir ~/.npm-global`
2. `npm config set prefix ~/.npm-global`
3. `echo "PATH=~/.npm-global/bin/:$PATH" >> .zprofile`

4. `echo "export PATH" >> .zprofile`
5. `source .zprofile`
6. `npm i -g @zowe/cli`

Note: if your default terminal shell is `bash` instead of `Zsh`, these instructions will need modified to update your `.bashrc` or `.bash_profile` file instead of the `.zprofile` file.

Once you have successfully installed the ZOWE CLI (or have reached a point where it is clear you cannot) then restart VSCode.

3 ZOWE NPM INSTALL FOR WINDOWS

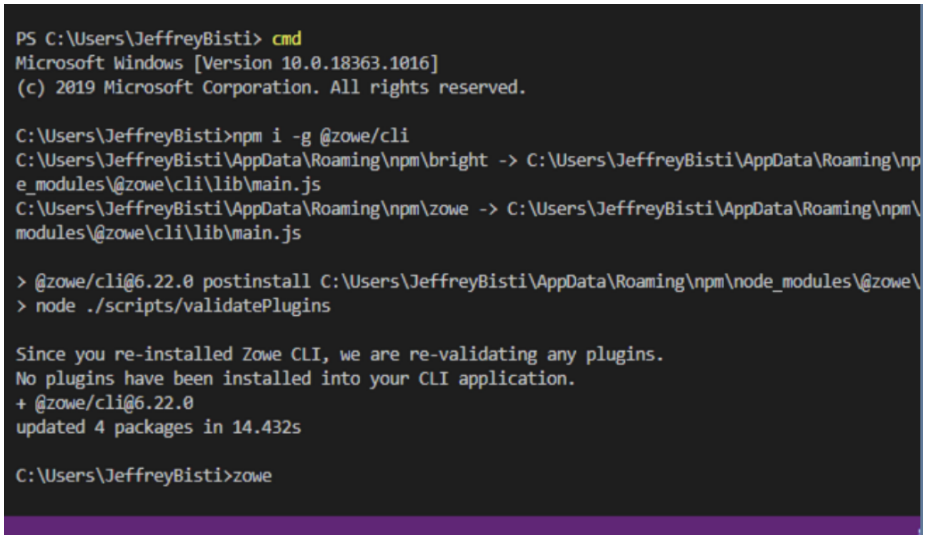
READ this whole section before installing the ZOWE CLI

First, exit from your VSCode application - make sure VSCode is not running.

On your Windows laptop or workstation, first switch to a CMD terminal instead of the default PowerShell terminal - then install Zowe CLI using 'npm', the Node Package Manager.

This should work for most users, though your output may be different than what you see in the screenshot.

1. type `cmd` (this will change the shell to CMD from `PowerShell`)
2. `npm i -g @zowe/cli`
3. `zowe`



If you see a response like "zowe : command not found", close the terminal window, and launch it again. Remember to switch to CMD.

If you still see a "command not found" error, you will need to update your PATH environment variable to include the location of the zowe program - usually in a "node_modules/bin" folder under C:\Windows, or under your home directory.

You can also try running the second command again but remove the '-g' option:

```
npm i @zowe/cli
```

Perform an internet search for how to set your Windows PATH variable as it varies from version to version, and can also be different if your laptop has been set up for roaming profiles and group policies.

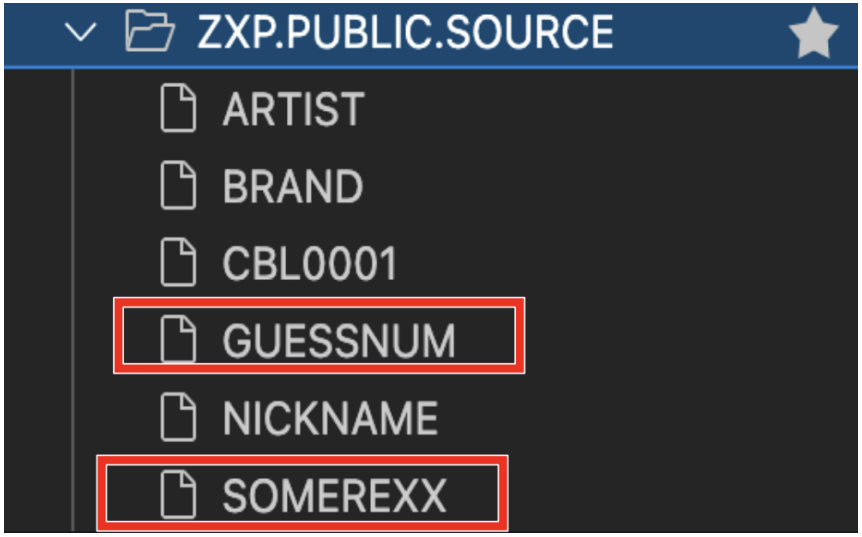
In general, if open-source commands like zowe and ssh don't work as expected, try running the command from a CMD terminal, instead of a PowerShell terminal. If the terminal window prompt in VSCode starts with PS > or similar, type CMD to get to an easier "shell" environment.

Once you have successfully installed the ZOWE CLI (or have reached a point where it is clear you cannot) then restart VSCode.

4 SOURCE AND TERMINAL

Get started on your REXX programming journey by copying two members from ZXP.PUBLIC.SOURCE into your own SOURCE data set.

Specifically, you are looking for SOMEREXX and GUESSNUM



Next, open up a Terminal window, either within VSCode or directly from your workstation/laptop operating system, just like you did for the USS challenge, but *DO NOT* use SSH to connect to the mainframe this time.

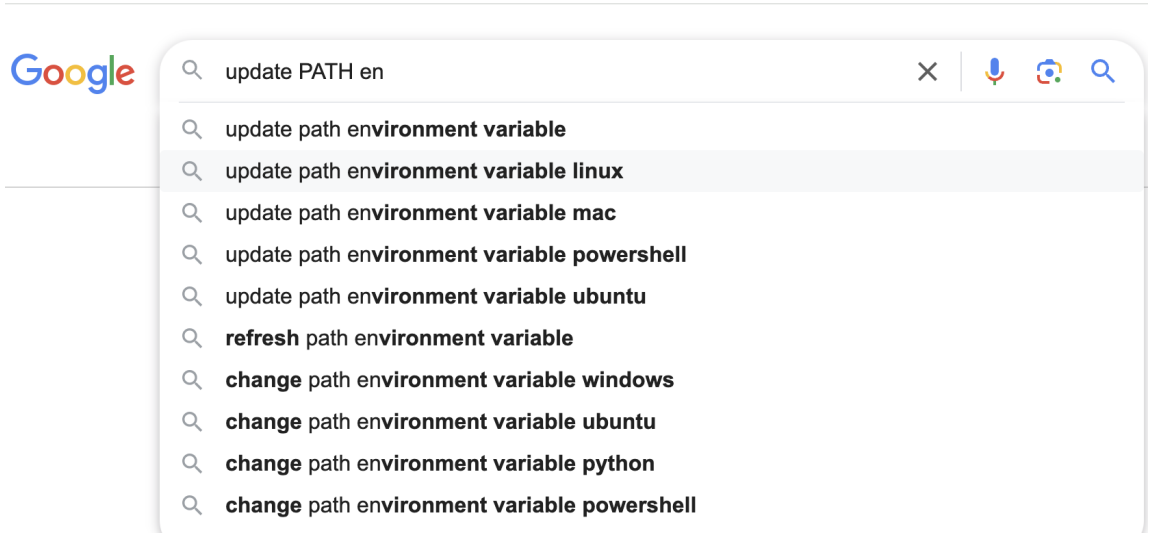
You should be able to type the command zowe from here and get some useful output from the program:

```
DESCRIPTION
 -----------

   Welcome to Zowe CLI!

   Zowe CLI is a command line interface (CLI) that provides a simple and
   streamlined way to interact with IBM z/OS.
```

If you see a message like “command not found”, you will need to locate where the zowe command is installed and add that location to your PATH environment variable. How this is done depends what operating system you are using. Use internet search for instructions that best match your laptop/workstation:



REXX1241112-1738

5 CREATE A TSO PROFILE

So far, in VSCode, you been using a z/OSMF profile to make connections to the mainframe

You may now also need to create a zowe TSO profile to issue TSO commands using the Zowe CLI.

5.1 USING TEAM CONFIGURATION FILES

Enter the following command to see what profiles are already defined:

```
zowe config list
```

You should see a print-out that shows your z/OSMF profile:

```
  zosmfZXPlore:
    type:       zosmf
    properties:
      port:               10443
      host:               204.90.115.200
      rejectUnauthorized: false
      user:               (secure value)
      password:           (secure value)
    secure:
      - user
      - password
```

and the basic TSO profile:

```
  tsoZXPlore:
    type:       tso
    properties:
```

```
      account:         FB3
      codePage:        1047
      logonProcedure: IZUFPROC
    secure:
      (empty array)
```

There should also be section at the end of the print-out that shows the default profiles:

```
defaults:
  zosmf: zosmfZXPlore
  tso:   tsoZXPLore
```

Your profile names may be different - make sure the names in the defaults section match your profiles.

REXX1241112-1738

6 SET DEFAULT PROFILES

At this point, you need to make sure the right profiles are set as defaults.

That means when you issue a zowe CLI command, you should not have to provide hostname, port, userid, password, etc when the command executes as all that information will be available from the profiles.

6.1 USING TEAM CONFIGURATION FILES

Your defaults should already be set but you can check using :

```
zowe config list defaults
```

which should show something similar to:

```
zosmf: zosmfZXPlore
tso:   tsoZXPlore
ssh:   ssh
```

If you have other profiles from other mainframe activities or connections, sometimes a restart of VSCode is required to make a profile switch take effect.

6.2 TEST YOUR DEFAULT CONNECTION

A quick test to make sure that your default profiles are working - execute a simple dataset list command:

```
zowe files list ds "ZXP.PUBLIC.*"
```

this should respond with :

```
ZXP.PUBLIC.CLIST
ZXP.PUBLIC.DATA
ZXP.PUBLIC.DB2.CBL
ZXP.PUBLIC.DB2.DATA
ZXP.PUBLIC.DB2.JCL
ZXP.PUBLIC.DB2.SQL
ZXP.PUBLIC.EXEC
ZXP.PUBLIC.GLOBAL.SMPPTS
ZXP.PUBLIC.INPUT
ZXP.PUBLIC.JCL
...
```

If instead you are prompted for any missing option - host, port, account number, etc - then your defaults profiles are not (yet) set up correctly.

Review the last couple of sections and make sure all the required settings are in place.

7 RUN YOUR FIRST REXX

Type the following command:

```
zowe tso issue command "exec 'Zxxxxx.SOURCE(somerexx)'" --ssm
```

You should receive a greeting message from the mainframe.

Make sure to include all of the double and single quotes.

If you receive requests for ‘account’, ‘hostname’, ‘userid’, or ‘password’, you almost certainly have not set the default profiles correctly - go back a couple of steps and correct as needed.

This may be the last time we point out that whenever you see Zxxxxx or Z99994, you need to input your own Z-userid.

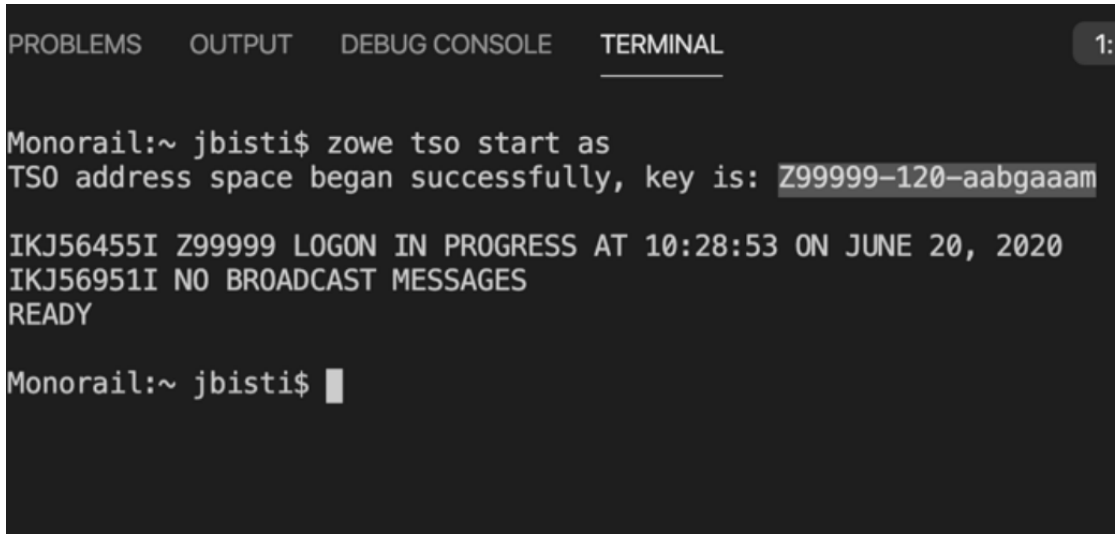
8 START AN ADDRESS SPACE

This time, instead of a single command, you will run an interactive REXX program; to make that work, you will need to create a server 'address space' to start and run the REXX command until you are finished.

Start an address space with the following command:

`zowe tso start as`

(here `as` is short for 'address-space')



This will create an address space for you, and tell you its **key**, which will begin with your userid (as you can see above).

You will need this key for the next few steps - it will be referred to in this document as "{my-as-key}".

Sometimes, after not being used for a while, a TSO address space goes away. If that happens, just make another one using the same `zowe tso start as` command - **take note of the new key it returns**.

You can choose to stop an address space with the command:

```
zowe tso stop as {my-as-key}
```

9 RUN THE SAME REXX

Run the same REXX program from step 7, but this time, direct the input towards the address space (and remember, you can probably press the Up arrow to recall previous commands)

zowe tso send as {my-as-key} --data "exec 'Zxxxxx.SOURCE(somerexx)'"

Notes:

1. That command is all on one line
2. {my-as-key} is the key from the address-space you just started
3. this time, leave off the '–ssm' option

You should get back the exact same response as before.

The big difference here is that you're now issuing these commands to a semi-persistent TSO Address Space, which will make more sense in the next step.

“TSO? ADDRESS SPACE?”

TSO (Time Sharing Option) is another way that z/OS allows many, many users to get access to data sets, run programs, and look at output. It is essentially the command-line interface for z/OS (when you’re not using USS to access the UNIX side of things).

Think of an ‘Address Space’ as a ticket that lets you start using system memory. An address space represents an enormous amount of memory, though the system will actually still control what lives in real on-chip memory, versus what gets moved (or paged) out to disk.

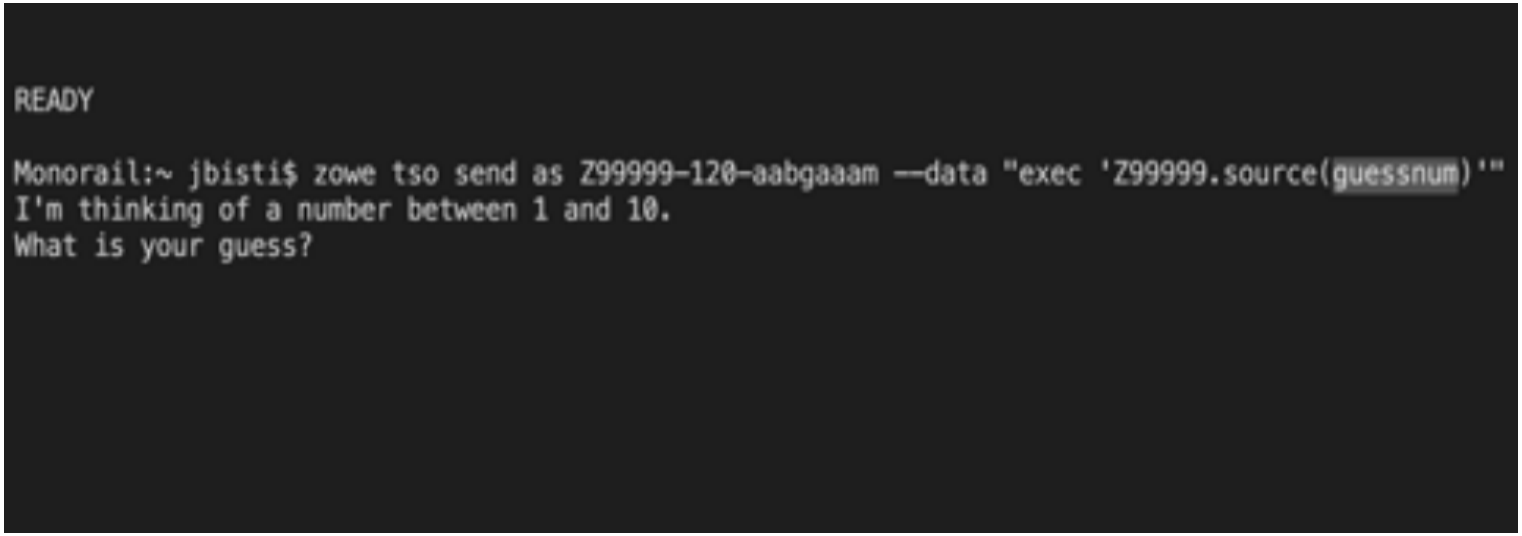
Where your program goes its memory from depends on how important it is, how it should be used, and if it will be shared with other programs.

Address Spaces are a core part of z/OS, and you should read more about them when you get a minute:

https://www.ibm.com/support/knowledgecenter/zosbasics/com.ibm.zos.zconcepts/zconcepts_82.htm

10 GUESS WHAT? ANOTHER EXEC

Assuming your TSO address space is active, run the program GUESSNUM using the same command (just change the member name from SOMEREXX to GUESSNUM).



This is a program that generates a random number between 1 and 10, and you have to guess that number.

There may be other games you'd rather be playing right about now, but not all of them will teach you about REXX and TSO, so keep that in mind if you're looking over at your XBOX!

11 SEE HOW CUTE THE CODE IS

If you haven't already done so, open up the code for that program in your VSCode editor.

It is not a complicated program by any means, but you might be noticing just how simple this REXX code really is.

16 lines of code, including a comment and a blank line; and yes, those "say" and "pull" commands really do what you think they do.

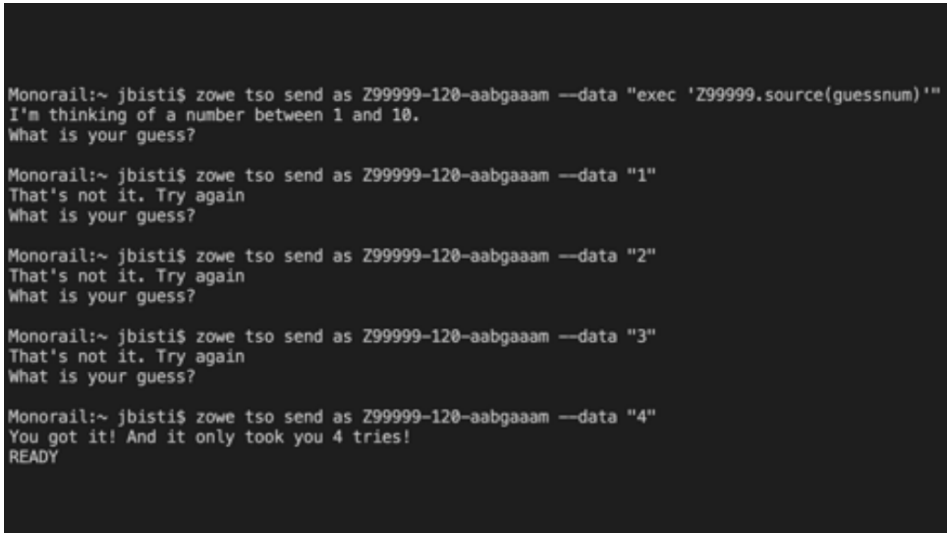
You can see why people love this language.

It's also got what I consider to be the world's greatest logo for a programming language.

Look at it. Just. Look. At. It ...

12 LIVE CHAT WITH REXX

Now you can send your guesses to the program by replacing everything between the double-quotes with a number.



You can see that the commands use the Address Space Key to ensure your input keeps going to the correct TSO address space, which is sitting there waiting for the next input.

If you get it right on the first try, congratulations!

Feel free to start the program again and make sure you can see it go through the “Try again” steps.

Now submit your completion check – **CHKREXX1** from **ZXP.PUBLIC.JCL**

Nice job - let's recap	Next up ...
Now you're getting into the swing of things. You're interacting with a Rexx program, running in a TSO address space, and you're doing that through the zowe command. You've probably also learned quite a bit about the Rexx language, and may have even done some extra reading about Address Spaces. Everything in here will help you become a more skilled mainframe professional.	The iron is most definitely smoldering, and you're probably becoming a fan of Rexx. For now, though, let's continue to the rest of the challenges in Fundamentals.