googled: list ip of current machine linux

first link: https://www.wikihow.com/Check-the-IP-Address-in-Linux

last method: run "hostname -I" in terminal

why last? because it gives me directly the IP address and I don't have to fish it out of some paragraph of text

man page for hostname says:

-I, --all-ip-addresses

Display all network addresses of the host. This option enumerates all configured addresses on all

network interfaces. The loopback interface and IPv6 link-local addresses are omitted.

I tried:

$ hostname -I

192.168.10.130 172.17.42.1

Here I got lucky because I have access to a computer which has more than one IP address configured and so I know what the output looks like. Seems like we get a bunch of IP adresses separated by space. Now we need to figure out:

1. How to test whether the output of this command is empty...

2. ... and how to reformat the output so that the IP addresses appear by one at a line.

The second part is easier for me because I already have some experience. There are many ways to do it, but there is this little program called tr which means translate and it translates characters to other characters. So let's try to translate space characters to newline characters!

$ hostname -I | tr " " "\n"

192.168.10.130

172.17.42.1

Yay! (But there seems to be a blank line at the end. Hmm... Hope it won't be a problem.)

OK. So let's deal with the first part. How to test whether the result of hostname -I is empty? We are testing output of a programm that we read in a terminal, so it's a job for shell. So I guess we first assign the output to some variable and then play with that.

$ ips=`hostname -I | tr " " "\n"`

$ echo $ips

192.168.10.130 172.17.42.1

Hmmm... How do we test whether it is empty? We better first make sure that the variable doesn't contain only the whitespace. I suspect that some extra whitespace was the result for the blank line.

google: bash variable strip whitespace

first link: https://stackoverflow.com/questions/369758/how-to-trim-whitespace-from-a-bash-variable

First answer seems a bit complicated but scrolling down I see something easy! Reroute through xargs!

$ hostname -I | xargs | tr " " "\n"

192.168.10.130

172.17.42.1

And now the last blank line is gone!

Yay!

Just to be sure, let's test what xargs does to just one whitespace.

$ echo " " | xargs | tr " " "W"

No "W", no whitespace. Can test the behaviour a bit more by

~$ echo " test " | tr " " "W"

WtestWW

~$ echo " test " | xargs | tr " " "W"

test

Good. Where were we? Ah! Now we have all IPs neatly trimmed from both sides in a varibale and we now need to check whether it is empty.

google: shell test variable empty

first link: https://stackoverflow.com/questions/3061036/how-to-find-whether-or-not-a-variable-is-empty-in-bash-script

Ugh. Confused as hell. They refer to bash, but I need just sh! Grrr. They mention "man test". Browsing that I see

-z STRING

the length of STRING is zero

Meh. Let's try. (Omit several unsuccessful tries which failed because I didn't know that -z has to be separated by a space from [.)

$ empty=""

$ if [ -z $empty ]; then echo "Empty variable"; else echo "Nonempty variable"; fi

Empty variable

It works? All that talk on stackoverflow is probably discussing differences between empty strings and unset variables or whatever. The man page is pretty clear. But does it work in sh? Or is this test only available in bash? Well. First let me test also the case of nonempty variable.

$ if [ -z $ips ]; then echo "Empty variable"; else echo "Nonempty variable"; fi

bash: [: 192.168.10.130: binary operator expected

Nonempty variable

Huh? Aha. I guess I should follow the suggestions given on stackoverflow more closely and put the variable name in double quotes.

$ if [ -z "$ips" ]; then echo "Empty variable"; else echo "Nonempty variable"; fi

Nonempty variable

There you go!

So what about this bash / sh (in)compatibility? Going back to the last google results and clicking on the third link gives answer that talks about "POSIX-compatible shell" I guess [ -z "${VAR}" ] has to be good enough.

OK. Now we just need to put all things together.

IPs=`hostname -I | xargs` # get all ip adresses with trimmed whitespace on both sides

if [ -z "${IPs}" ]; then

echo "I am lost"

else

echo $IPs | tr " " "\n"

fi