Run and debug Intel assembly code with Eclipse on Linux

1. Download the "print_args64.asm" assembly example from http://ti.uni-bielefeld.de/html/teaching/WS1819/techinf1/index.html. If you are on a 32 bit machine, choose "print_args.asm" instead.
2. The following steps aim to create an Eclipse project that uses a makefile to assemble and link the desired program. A makefile consists of directives to automatically generate specific targets, for example by assembling and linking. To generate machine code for print_args64, it is sufficient to create a new file called "Makefile", containing the makefile code below. Check the formatting! Place the newly created makefile next to "print_args64.asm".

```
ASM_NAME = print_args64

all: $(ASM_NAME)
clean:
	rm -f $(ASM_NAME).o $(ASM_NAME)

$(ASM_NAME): $(ASM_NAME).o
	ld -m elf_x86_64 -static -o $(ASM_NAME) $(ASM_NAME).o

$(ASM_NAME).o: $(ASM_NAME).asm
	nasm -f elf64 -F dwarf -g $(ASM_NAME).asm
```

3. If you are using your own computer rather than GZI workstations, you might need to install "nasm" and "ld" if you have not already (e.g. `sudo apt install nasm binutils` for Ubuntu).
4. Check your makefile by opening a terminal, changing the working directory to the location of your source code/makefile, then running "make". If an executable with your desired name is created, proceed.
5. Download **Eclipse IDE for C/C++ Developers** from https://www.eclipse.org/downloads/packages/. Extract the archive, then run the eclipse executable. If you are getting a "permission denied", use `chmod u+x eclipse` to change the rights appropriately.
6. Choose any directory except the one that contains your source code as your workspace – the default one will do.
7. From the welcome screen, select "Import a project with a working makefile" (if you closed the welcome screen and want to restore it, go to "Help" -> "Welcome"). Continue by choosing a project name and the location of the existing code (for example with "Browse..."), then click "Finish".
8. Close the welcome screen.
9. If you did not build an executable in step 4, do it now by clicking the build button (small hammer) in Eclipse or running "make" in a terminal as described above.
10. Create a new launch configuration by clicking on the dropdown menu currently stating "No Launch Configurations", then selecting "New Launch Configuration...". Configure as follows:
    - Launch Mode: "Debug"
    - Launch Configuration Type: "C/C++ Application"

    - Click "Next"
    - Project: *Choose any project name here*
    - C/C++ Application: "print_args64" (or the name you chose for your executable)
    - Click "Finish"
11. Open "print_args64.asm" by double-clicking it in Eclipse's Project Explorer on the left hand side, then add a breakpoint at your desired location in the program by double clicking on the line number in the editor.
12. Switch the Launch Mode to "Debug" (drop-down menu to the left of the Launch Configuration drop-down), then run your Launch Configuration. If Eclipse asks you if you would like to switch to the Debug perspective, agree by choosing "Switch". The program should execute up to your breakpoint.
13. To inspect CPU register entries, click the "Registers" tab, which per default resides in the lower part of the editor window. If it does not exist, go to "Window", "Show View", then either select "Registers" there or choose "Other...", then type "Registers" as the filter text.
14. To inspect memory regions, click the "Memory" tab, then the green "+". Now enter a memory location and click OK. Eclipse shows hexadecimal values per default, but you can change the interpretation of the data stored in memory by choosing the Button "New Renderings..." and then choosing a rendering (e.g. ASCII) by double clicking it.
15. To change the arguments your program starts with, click the cogwheel next to your Launch Configuration in the dropdown menu. In the new window, select the "Arguments" tab and add your desired arguments in the text field.