1. Installing R software (Linux)

Execute the following in a terminal to install the R interpreter:

```
sudo apt update -qq
# install two helper packages we need
sudo apt install --no-install-recommends software-properties-common
dirmngr
# add the signing key (by Michael Rutter) for these repos
# To verify key, run gpg --show-keys
/etc/apt/trusted.gpg.d/cran_ubuntu_key.asc
# Fingerprint: E298A3A825C0D65DFD57CBB651716619E084DAB9
wget -qO- https://cloud.r-
project.org/bin/linux/ubuntu/marutter_pubkey.asc | sudo tee -a
/etc/apt/trusted.gpg.d/cran_ubuntu_key.asc
# add the R 4.0 repo from CRAN -- adjust 'focal' to 'groovy' or 'bionic'
as needed
sudo add-apt-repository "deb https://cloud.r-
project.org/bin/linux/ubuntu $(lsb_release -cs)-cran40/"

sudo apt install --no-install-recommends r-base

sudo add-apt-repository ppa:c2d4u.team/c2d4u4.0+
```

Now we can install R Studio:

```
wget https://download1.rstudio.org/electron/jammy/amd64/rstudio-
2023.12.1-402-amd64.deb -O rstudio-2023.12.1-402-amd64.deb && \
sudo chmod +x rstudio-2023.12.1-402-amd64.deb && \
sudo dpkg -i rstudio-2023.12.1-402-amd64.deb
```

PROF

`R studio` will automatically create a `.Rprofile` file in your `home` directory.
We want to add the code hook into that environment file.

```
local({ x <- options()$repos
if (!is.element("CRAN", x)) {
  x["CRAN"] = "@CRAN@"
}
x["SimpactCyan"] <- "http://research.edm.uhasselt.be/jori"
options(repos = x) })
```

Now we need to source ourself in the R terminal with:

```
source("~/.Rprofile")
```

Then we can install the package:

```
install.packages("RSimpactCyan")
```

follow the prompts and select a server mirror(e.g Belgium)

Now in R studio's terminal you can call:

```
library(RsimpactCyan)
```

2. Installing C++ software (Linux)

Install the following system packages:

```
sudo apt-get install build-essential
sudo apt-get install libgsl-dev
sudo apt-get install libtiff-dev
```

Building SimpactCyan from Source (NOT WORKING FULLY):

Now we have to:
2.1 Clone the project into your home directory.
2.2 Create a build folder inside our cloned project.
2.3 CD into that build folder
2.4 Run the following commands:


```
cmake ..
make -j 4
./simpact-cyan-release
```

That's it, you've now built the project from source. For troubleshooting check out the code for building the package from source in `install-R.sh`.

2.5 Build Debian package (WORKING)

execute the following python file:

```
```

```
python3 buildscripts/build-debian.py .
```

That command will build the debian package and deposit it under the `/tmp/` folder in root `/`.

For compiling C++ code you can execute the following in the terminal:

```
g++ -o over overriddetest.cpp
```

In the above example `over` is the name that we are assigning to our program. You can call you program whatever you'd like. `overriddetest.cpp` is the name of the C++ file that we are compiling.
After compilation we can execute our C++ program like this:

```
./over
```