

Made by: Tzvi Mints, Or Abuhazira, Eilon Tsadok

Created during a computer communication course during the second year at Ariel University in the Department of Computer Science, 2019

Purpose

Run this program on your executable file in order to find if it compile successfully, have no memory leaks, have no problem with the thread race.

How to use

Run this program from the terminal. such as the first argument is the path of the make file, the second argument is the name of the executable file, and the third argument is arguments if the program need for the running.

Possible outputs

Photo	State
<pre> eilon26@eilon26-VirtualBox:~/Documents\$./BasicCheck.sh ./FolderWithoutMakeFile output BasicCheck.sh ./FolderWithoutMakeFile <output> Makefile Not Found Compilation Memory leaks thread race FAIL FAIL FAIL Summary: 7 </pre>	Folder without Makefile
<pre> eilon26@eilon26-VirtualBox:~/Documents\$./BasicCheck.sh ./Check output BasicCheck.sh ./Check <output> Please 100 Compilation Memory leaks thread race PASS FAIL PASS Summary: 2 </pre> <pre> #include <iostream> #include <stdlib.h> using namespace std; int main() { cout << "Please 100 \n"; char* p = (char*)malloc(555); return 0; } </pre>	Memory leak
<pre> eilon26@eilon26-VirtualBox:~/Documents\$./BasicCheck.sh ./Check output BasicCheck.sh ./Check <output> Please 100 Compilation Memory leaks thread race PASS PASS PASS Summary: 0 </pre> <pre> #include <iostream> #include <stdlib.h> using namespace std; int main() { cout << "Please 100 \n"; char* p = (char*)malloc(555); free(p); return 0; } </pre>	Normal
<pre> eilon26@eilon26-VirtualBox:~/Documents\$./BasicCheck.sh ./Check NoExistFile BasicCheck.sh ./Check <NoExistFile> Compilation Memory leaks thread race FAIL FAIL FAIL Summary: 7 </pre>	No Exists File

<pre> eilon26@eilon26-VirtualBox:~/Documents\$./BasicCheck.sh ./Check output BasicCheck.sh ./Check <output> Compilation Memory leaks thread race PASS PASS FAIL Summary: 1 </pre>	Thread race
---	-------------

Script explanation

Photo	Explanation
<pre> # This function is responsible to compile the input # Program and then go to step 3, which is memory check compile() { ./\$1 \$2 2>/dev/null if [\$? -eq 0] then Answer[0]=PASS return 0 else return 1 fi } </pre>	This function is responsible to compile the input. Program and then go to step 3, which is memory check
<pre> # This function is responsible to check for memory leaks used by Valgrind memorychk() { valgrind --leak-check=full --error-exitcode=1 ./\$1 \$2 >/dev/null 2>&1 if [\$? -eq 0] then Answer[1]=PASS return 0 else return 1 fi } </pre>	This function is responsible to check for memory leaks used by Valgrind
<pre> #!/bin/bash Answer=(FAIL FAIL FAIL) # This function is responsible for print output to the screen output_to_screen() { echo "Compilation Memory leaks thread race" echo -e " \${Answer[0]} \t\t \${Answer[1]} \t\t \${Answer[2]}" echo -e "\t\t Summary: \$1" exit \$1 } </pre>	This function is responsible for print output to the screen
<pre> # This function is responsible for making step 3 in the assignment step3() { compile \$1 \$2 first=\$? memorychk \$1 \$2 second=\$? threaddebugger \$1 \$2 third=\$? answer=\$((2#\$first\$second\$third)) output_to_screen \$answer } </pre>	This function is responsible for making step 3 in the assignment

```
# This function is responsible for Thread debugger used by Helgrind
threaddebugger() {
    valgrind --tool=helgrind ./$1 $2 >/dev/null 2>&1
    if [ $? -eq 0 ]
    then
        Answer[2]=PASS
        return 0
    else
        return 1
    fi
}
```

This function is responsible for Thread debugger used by Helgrind

```
# Search for Makefile
cd $dir_path
find Makefile >/dev/null 2>&1
if [ $? -eq 0 ]
then
    make >/dev/null 2>&1
    if [ $? -eq 0 ]
    then
        step3 $program $arguments
    else
        output_to_screen 7
    fi
else
    echo "Makefile Not Found"
    output_to_screen 7
fi
```

Search for Makefile

```
# Check if there current amount of values
if [ $# -lt 2 ] # Less then 2
then
    echo "There Less Then 2 Arguments"
    exit 7
fi

dir_path=$1
program=$2
shift 2
arguments=$@

# Print First Line
echo "BasicCheck.sh $dir_path <$program> $arguments"
```

Check if there current amount of values