# Submission Report

* Submission generated at 09/30/2025 at 17:41:14
* Machine info: Darwin mac.boisestate.edu 24.6.0 Darwin Kernel Version 24.6.0: Mon Jul 14 11:30:40 PDT 2025; root:xnu-11417.140.69~1/RELEASE\_ARM64\_T8132 arm64

## Note to Students

Please read this report carefully before submission. Ensure that all sections are complete and accurate. Look for any errors in the build or test outputs. If you find any issues, correct them before submitting. Post any questions on the class discussion board for help.

## README

# Project X

* Name: John Doe
* Email: johndoe@u.boisestate.edu
* Class: 452-001

## Known Bugs or Issues

TODO: Are there any known issues?

## Experience

TODO: Describe your experience with the project (struggles, breakthroughs, etc.).

## Analysis

TODO: Provide your analysis of the results. If the assignment does not require analysis, you can remove this section.

Here is an example of how to include a plot in your README:

## Example Image

## Build Output

This section was generated by running make all in the project root directory.

mkdir -p build/debug  
cc -g -O0 -DDEBUG -fno-omit-frame-pointer -fsanitize=address -c src/lab.c -o build/debug/lab.c.o  
mkdir -p build/debug  
cc -g -O0 -DDEBUG -fno-omit-frame-pointer -fsanitize=address -c src/main.c -o build/debug/main.c.o  
cc -g -O0 -DDEBUG -fno-omit-frame-pointer -fsanitize=address build/debug/lab.c.o build/debug/main.c.o -o build/debug/myapp\_d -fsanitize=address  
mkdir -p build/release  
cc -Wall -Wextra -O2 -fPIE -MMD -MP -Wformat -Wformat=2 -Wconversion -Wsign-conversion -Wimplicit-fallthrough -fstack-protector-strong -Werror=format-security -Werror=implicit -Werror=incompatible-pointer-types -Werror=int-conversion -c src/lab.c -o build/release/lab.c.o  
mkdir -p build/release  
cc -Wall -Wextra -O2 -fPIE -MMD -MP -Wformat -Wformat=2 -Wconversion -Wsign-conversion -Wimplicit-fallthrough -fstack-protector-strong -Werror=format-security -Werror=implicit -Werror=incompatible-pointer-types -Werror=int-conversion -c src/main.c -o build/release/main.c.o  
cc -Wall -Wextra -O2 -fPIE -MMD -MP -Wformat -Wformat=2 -Wconversion -Wsign-conversion -Wimplicit-fallthrough -fstack-protector-strong -Werror=format-security -Werror=implicit -Werror=incompatible-pointer-types -Werror=int-conversion build/release/lab.c.o build/release/main.c.o -o build/release/myapp   
mkdir -p build/tests  
cc -g -O0 -DTEST -fprofile-arcs -ftest-coverage -c src/lab.c -o build/tests/lab.c.o  
mkdir -p build/tests  
cc -g -O0 -DTEST -fprofile-arcs -ftest-coverage -c src/main.c -o build/tests/main.c.o  
mkdir -p build/tests/harness/  
cc -g -O0 -DTEST -fprofile-arcs -ftest-coverage -c tests/harness/unity.c -o build/tests/harness/unity.c.o  
mkdir -p build/tests/  
cc -g -O0 -DTEST -fprofile-arcs -ftest-coverage -c tests/lab-test.c -o build/tests/lab-test.c.o  
cc -g -O0 -DTEST -fprofile-arcs -ftest-coverage build/tests/lab.c.o build/tests/main.c.o build/tests/harness/unity.c.o build/tests/lab-test.c.o -o build/tests/myapp\_t -fprofile-arcs -ftest-coverage  
mkdir -p build/debug-test  
cc -g -O0 -DDEBUG -DTEST -fno-omit-frame-pointer -fsanitize=address -c src/lab.c -o build/debug-test/lab.c.o  
mkdir -p build/debug-test  
cc -g -O0 -DDEBUG -DTEST -fno-omit-frame-pointer -fsanitize=address -c src/main.c -o build/debug-test/main.c.o  
mkdir -p build/debug-test/harness/  
cc -g -O0 -DDEBUG -DTEST -fno-omit-frame-pointer -fsanitize=address -c tests/harness/unity.c -o build/debug-test/harness/unity.c.o  
mkdir -p build/debug-test/  
cc -g -O0 -DDEBUG -DTEST -fno-omit-frame-pointer -fsanitize=address -c tests/lab-test.c -o build/debug-test/lab-test.c.o  
cc -g -O0 -DDEBUG -DTEST -fno-omit-frame-pointer -fsanitize=address build/debug-test/lab.c.o build/debug-test/main.c.o build/debug-test/harness/unity.c.o build/debug-test/lab-test.c.o -o build/debug-test/myapp\_td -fsanitize=address  
Builds completed. You can run the application with: ./build/release/myapp  
You can run the debug build with: ./build/debug/myapp\_d  
You can run the test build with: ./build/tests/myapp\_t  
You can run the debug-test build with: ./build/debug-test/myapp\_td

## Coverage Report

This section was generated by running make report in the project root directory.

Setting up tests...  
Tearing down tests...  
tests/lab-test.c:32:test\_get\_greeting:PASS  
  
-----------------------  
1 Tests 0 Failures 0 Ignored   
OK  
./build/tests/myapp\_t  
Setting up tests...  
Tearing down tests...  
tests/lab-test.c:32:test\_get\_greeting:PASS  
  
-----------------------  
1 Tests 0 Failures 0 Ignored   
OK  
mkdir -p ./build/report/html  
mkdir -p ./build/report/txt  
gcovr -r . --html --html-details --exclude-directories build/tests/harness --exclude '.\*main\.c$' --exclude '.\*test\.c$' -o ./build/report/html/coverage\_report.html  
(INFO) Reading coverage data...  
(INFO) Writing coverage report...  
gcovr -r . --txt --exclude-directories build/tests/harness --exclude '.\*main\.c$' --exclude '.\*test\.c$'  
(INFO) Reading coverage data...  
(INFO) Writing coverage report...  
------------------------------------------------------------------------------  
 GCC Code Coverage Report  
Directory: .  
------------------------------------------------------------------------------  
File Lines Exec Cover Missing  
------------------------------------------------------------------------------  
src/lab.c 9 9 100%  
------------------------------------------------------------------------------  
TOTAL 9 9 100%  
------------------------------------------------------------------------------

## Address Sanitizer Report

This section was generated by running make leak-test in the project root directory.

==68281==AddressSanitizer: detect\_leaks is not supported on this platform.  
/bin/bash: line 1: 68281 Abort trap: 6 ASAN\_OPTIONS="detect\_leaks=1" ./build/debug-test/myapp\_td  
make: \*\*\* [leak-test] Error 134

## Src Files

### lab.c

#include "lab.h"  
#include <stdio.h>  
#include <stdlib.h>  
  
char \*get\_greeting(const char \*restrict name)  
{  
 if (name == NULL)  
 {  
 return NULL;  
 }  
  
 // Allocate memory for the greeting message  
 int length = snprintf(NULL, 0, "Hello, %s!", name);  
 if (length < 0) // GCOVR\_EXCL\_START  
 {  
 return NULL; // snprintf failed  
 } // GCOVR\_EXCL\_STOP  
  
 //Casting is safe here because we know length is non-negative  
 size\_t alloc\_size = (size\_t) length + 1; // +1 for the null terminator  
 char \*greeting = malloc( alloc\_size);  
  
  
 if (greeting == NULL) // GCOVR\_EXCL\_START  
 {  
 return NULL; // Memory allocation failed  
 } // GCOVR\_EXCL\_STOP  
  
  
 // Create the greeting message  
 snprintf(greeting, alloc\_size, "Hello, %s!", name);  
  
 return greeting;  
}

### lab.h

#ifndef LAB\_H  
#define LAB\_H  
  
/\*\* \* @brief Returns a greeting message.  
 \*  
 \* This function returns a string that contains a greeting message.  
 \* The string is allocated with malloc and should be freed by the caller.  
 \* @param name The name to include in the greeting.  
 \* @return A greeting string.  
 \*/  
char\* get\_greeting(const char\* restrict name);  
  
  
#endif // LAB\_H

### main.c

#include "lab.h"  
#include <stdio.h>  
#include <stdlib.h>  
  
#ifdef TEST  
#define main main\_exclude  
#endif  
  
  
  
int main(void)  
{  
 char \*greeting = get\_greeting("World");  
 if (greeting) {  
 printf("%s\n", greeting);  
 free(greeting); // Free the allocated memory for the greeting  
 } else {  
 printf("Failed to create greeting.\n");  
 }  
 return 0;  
}

## Tests Files

### lab-test.c

#include <stdlib.h>  
#include <stdio.h>  
#include "harness/unity.h"  
#include "../src/lab.h"  
  
  
void setUp(void) {  
 printf("Setting up tests...\n");  
}  
  
void tearDown(void) {  
 printf("Tearing down tests...\n");  
}  
  
void test\_get\_greeting(void) {  
 char \*greeting = get\_greeting("Alice");  
 TEST\_ASSERT\_NOT\_NULL(greeting);  
 TEST\_ASSERT\_EQUAL\_STRING("Hello, Alice!", greeting);  
 free(greeting); // Free the allocated memory for the greeting  
  
 greeting = get\_greeting(NULL);  
 TEST\_ASSERT\_NULL(greeting);  
  
 greeting = get\_greeting("");  
 TEST\_ASSERT\_NOT\_NULL(greeting);  
 TEST\_ASSERT\_EQUAL\_STRING("Hello, !", greeting);  
 free(greeting);  
}  
  
int main(void) {  
 UNITY\_BEGIN();  
 RUN\_TEST(test\_get\_greeting);  
 return UNITY\_END();  
}

## Scripts Files

Report generated on 09/30/2025 at 17:41:18

## End of Report

SHA-256 Hash of the report: e1ffdff024780271a3230891b152f327d21b7670f522a4af8a0dd9192bdba0b0

Do not edit the generated report. Any changes will be reported as academic dishonesty