Lab Report 1: Buffer Overflow Demonstration

Due: 10/17/2024 at 10pm

Adrian Nelson Aidan Chin

Code

What is Buffer Overflow? When does it occur?

Buffer Overflow is when the amount of data exceeds the storage capacity of a buffer (in our case, the student_id_1 and student_id_2 arrays of type char and length 9). Using an instruction like `scanf()`, we can store a sequence of characters of longer than a length of 9 to one of the (student_id_*) buffers. Using instructions like `scanf()`, `sprintf()`, `strcpy()`, etc. lead to possible buffer overflow by having no safeguards.

How can buffer overflow be prevented in a program?

There are a few protections that can be put in place, such as defining the size of the buffer when performing a read/reformat of characters. This is why coders like using `scanf("%9s", student_id_*)`, despite our buffers being specified as length 9, certain C instructions do not have protections in place to prevent buffer overflow, which is why it can be exploited as shown in this Lab.

What would be a good example of a buffer overflow attack?

A classic example of a buffer overflow attack is the classic 1988 Morris Worm exploit, which targeted the finger daemon in the Unix system. The finger daemon was a network query for user data, and it had a buffer that could be overfilled, with these overflows the Morris Worm was able to overwrite critical memory locations like return addresses and inject his own code to be run, resulting in a major security violation with code injection.

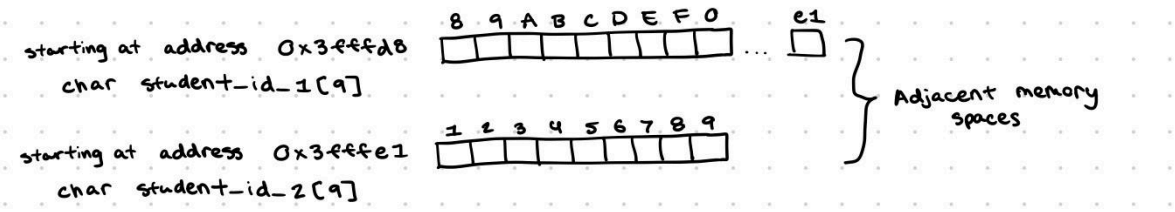
Explain how buffer overflow occurs in your code using screenshots of the code as well as the results shown on the terminal:

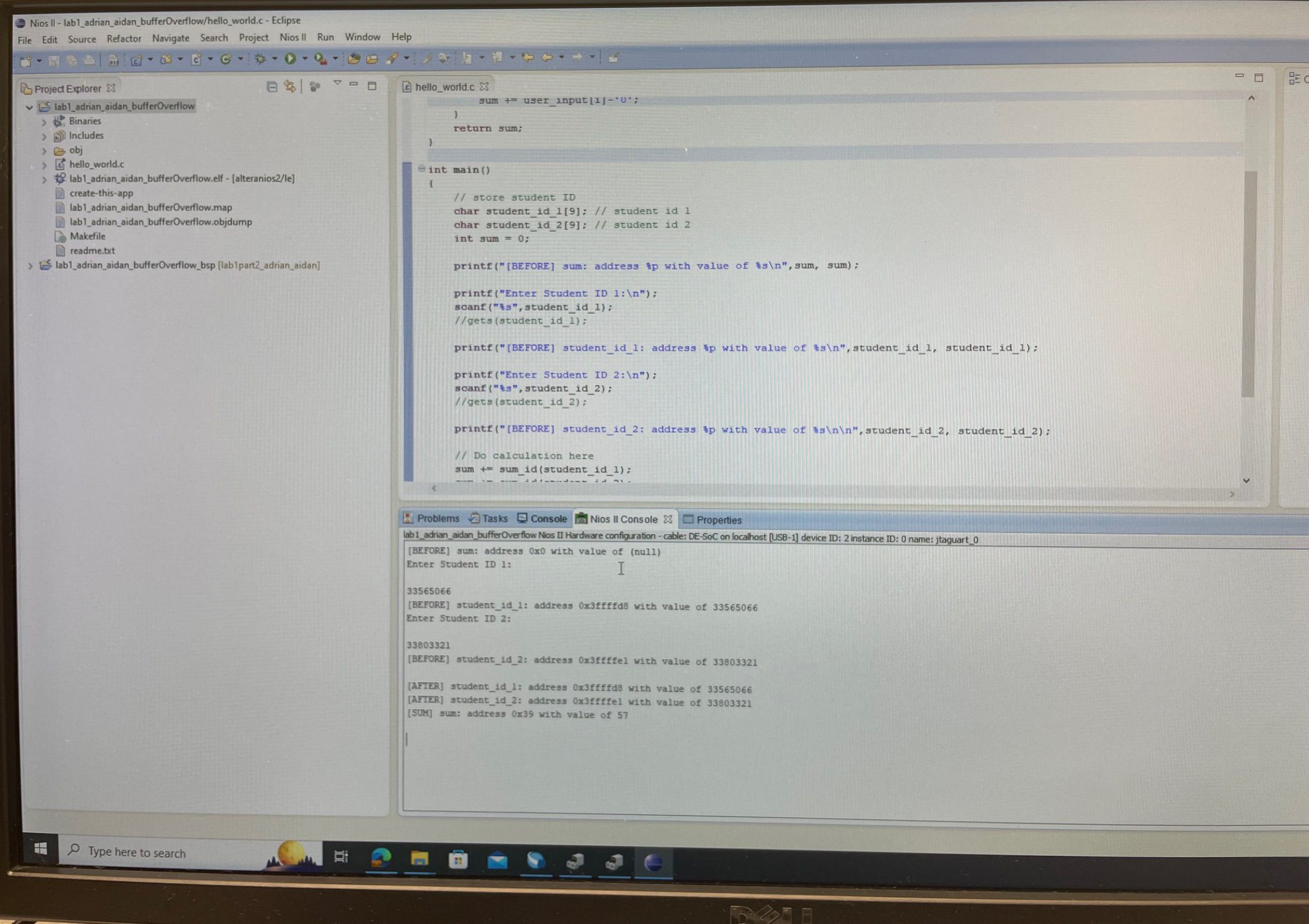
In our code, the `scanf("%s", student_id_1)` call can take in a char array of a length greater than 9 char (8 char and the `NULL` terminator `'\0'`, note that `"%9s"` would prevent buffer overflow using `scanf`) and less than 18 chars (unless we also buffer overflow `student_id_2` using the same method), causing memory space overlap between `student_id_1` and `student_id_2`. Since we wrote `student_id_1` first, the overflowed contents get overwritten when `student_id_2` gets written. The behavior we witness when printing the final values of `student_id_1` and `student_id_2`, is that `student_id_1` will show the contents of `student_id_2`, because the `student_id_1`'s `NULL` terminator was overwritten leading to `student_id_1` terminating at `student_id_2's NULL` terminator.

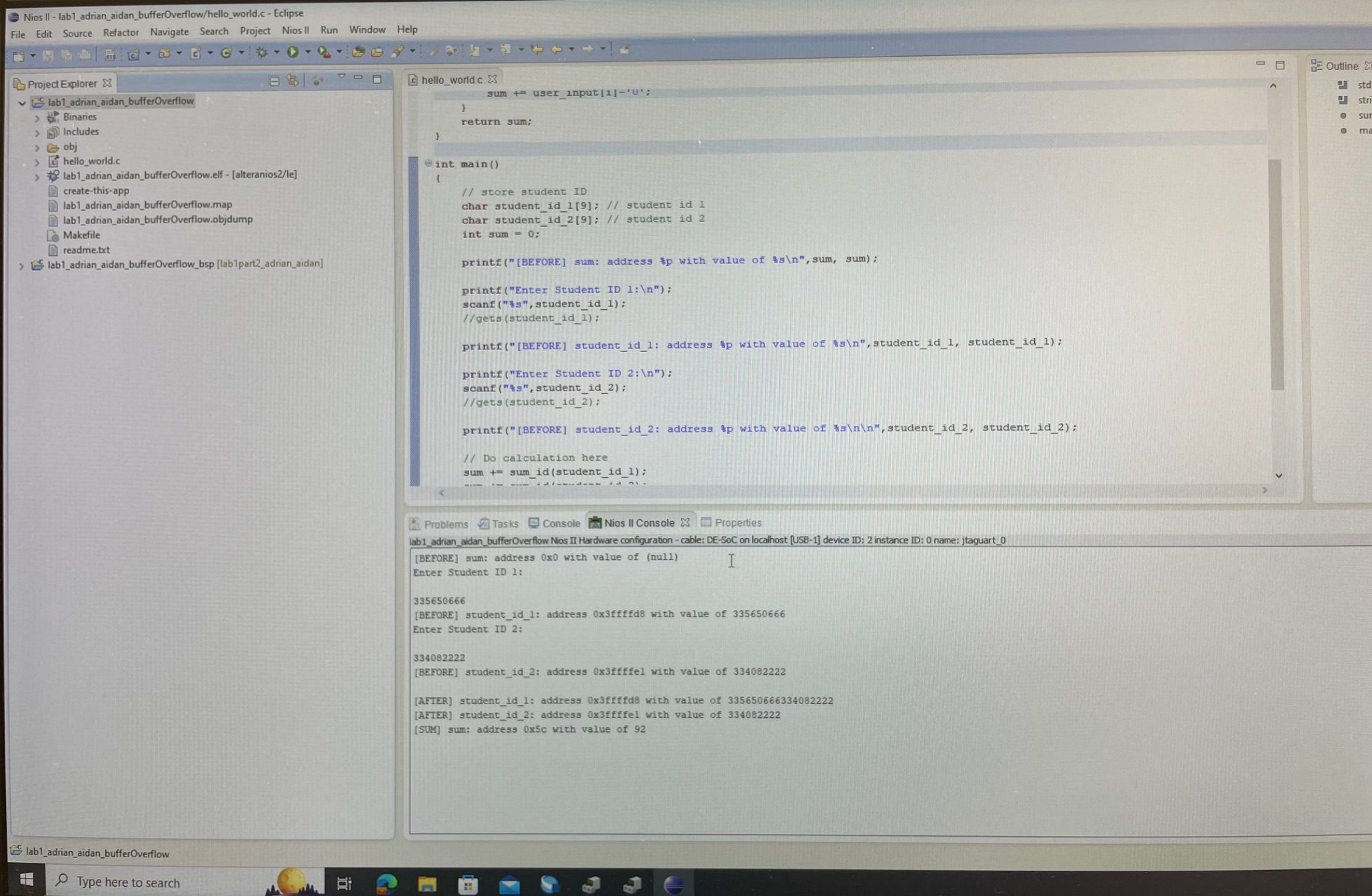
In our first image, we see expected behavior using student IDs of length 8 (and the `NULL` terminator). The addition using the sum function call comes out to 57, which is correct for the sum variable.

In our second image, we see unexpected behavior and buffer overflow occurring for both `student_id_1` and `student_id_2`, as they each have a length of 9 (and the `NULL` terminator). At the end of the program, you can clearly see that `student_id_1` is terminating at `student_id_2's NULL` terminator. That is a visual example of buffer overflow, showing you contents of a memory space that was unexpected due to `scanf` not having a safeguard/limit against getting an array larger than the prespecified buffer.

The page below shows the memory spaces:




Nios II - lab1_adrian_aidan_bufferOverflow/hello_world.c - Eclipse
File Edit Source Refactor Navigate Search Project Nios II Run Window Help
Project Explorer
lab1_adrian_aidan_bufferOverflow
Binaries
Includes
obj
hello_world.c
lab1_adrian_aidan_bufferOverflow.elf - [alteranios2/le]
create-this-app
lab1_adrian_aidan_bufferOverflow.map
lab1_adrian_aidan_bufferOverflow.objdump
Makefile
readme.txt
lab1_adrian_aidan_bufferOverflow_bsp [lab1part2_adrian_aidan]
hello_world.c
sum += user_input[i]-'0';
}
return sum;
}
int main()
{
// store student ID
char student_id_1[9]; // student id 1
char student_id_2[9]; // student id 2
int sum = 0;
printf("[BEFORE] sum: address %p with value of %s\n",sum, sum);
printf("Enter Student ID 1:\n");
scanf("%s",student_id_1);
//gets(student_id_1);
printf("[BEFORE] student_id_1: address %p with value of %s\n",student_id_1, student_id_1);
printf("Enter Student ID 2:\n");
scanf("%s",student_id_2);
//gets(student_id_2);
printf("[BEFORE] student_id_2: address %p with value of %s\n\n",student_id_2, student_id_2);
// Do calculation here
sum += sum_id(student_id_1);
Problems Tasks Console Nios II Console Properties
lab1_adrian_aidan_bufferOverflow Nios II Hardware configuration - cable: DE-SoC on localhost [USB-1] device ID: 2 instance ID: 0 name: jtaguart_0
[BEFORE] sum: address 0x0 with value of (null)
Enter Student ID 1:
33565066
[BEFORE] student_id_1: address 0x3ffffd8 with value of 33565066
Enter Student ID 2:
33803321
[BEFORE] student_id_2: address 0x3ffffe1 with value of 33803321
[AFTER] student_id_1: address 0x3ffffd8 with value of 33565066
[AFTER] student_id_2: address 0x3ffffe1 with value of 33803321
[SUM] sum: address 0x39 with value of 57
Type here to search


Nios II - lab1_adrian_aidan_bufferOverflow/hello_world.c - Eclipse
File Edit Source Refactor Navigate Search Project Nios II Run Window Help
Project Explorer
lab1_adrian_aidan_bufferOverflow
Binaries
Includes
obj
hello_world.c
lab1_adrian_aidan_bufferOverflow.elf - [alteranios2/le]
create-this-app
lab1_adrian_aidan_bufferOverflow.map
lab1_adrian_aidan_bufferOverflow.objdump
Makefile
readme.txt
lab1_adrian_aidan_bufferOverflow_bsp [lab1part2_adrian_aidan]
hello_world.c
sum += user_input[i]-'0';
}
return sum;
}
int main()
{
// store student ID
char student_id_1[9]; // student id 1
char student_id_2[9]; // student id 2
int sum = 0;
printf("[BEFORE] sum: address %p with value of %s\n",sum, sum);
printf("Enter Student ID 1:\n");
scanf("%s",student_id_1);
//gets(student_id_1);
printf("[BEFORE] student_id_1: address %p with value of %s\n",student_id_1, student_id_1);
printf("Enter Student ID 2:\n");
scanf("%s",student_id_2);
//gets(student_id_2);
printf("[BEFORE] student_id_2: address %p with value of %s\n\n",student_id_2, student_id_2);
// Do calculation here
sum += sum_id(student_id_1);
Outline
Problems Tasks Console Nios II Console Properties
lab1_adrian_aidan_bufferOverflow Nios II Hardware configuration - cable: DE-SoC on localhost [USB-1] device ID: 2 instance ID: 0 name: jtaguart_0
[BEFORE] sum: address 0x0 with value of (null)
Enter Student ID 1:
335650666
[BEFORE] student_id_1: address 0x3ffffd8 with value of 335650666
Enter Student ID 2:
334082222
[BEFORE] student_id_2: address 0x3ffffe1 with value of 334082222
[AFTER] student_id_1: address 0x3ffffd8 with value of 335650666334082222
[AFTER] student_id_2: address 0x3ffffe1 with value of 334082222
[SUM] sum: address 0x5c with value of 92
lab1_adrian_aidan_bufferOverflow
Type here to search