LW 9.2: Pass By Reference

Objectives

- To understand that:
  - argument passing is similar to initialization,
  - when a function is called, each formal argument is initialized by its corresponding actual argument, and
  - when a reference is provided as a formal parameter, the corresponding actual argument will be referred to by the formal parameter.

Labwork

- There are questions throughout this document that you will need to respond to for credit.
- There are questions at the end of this document that you will respond to for credit.
- Using Putty (PC) or terminal (Mac), log-in to `compute.cse.tamu.edu`
- Create an empty directory for this labwork. In a terminal (e.g. putty) navigate to this directory.
- Download the source code from https://drive.google.com/open?id=0B_ouNNuWgNZCVm5udjlUaXpBV1E and move it into a directory in your h:drive
- Verify you copied the file with the following command:
  ```
  ls
  ```
- Compile using the following command:
  ```
  g++-7.2.0 -std=c++17 -Wall -Wextra -pedantic -fsanitize=address,undefined
  *.cpp
  ```
- Inspect the source code; notice that the source code is *identical* to that in **LW 9.1: Pass By Value**, with the exception that the type of formal parameter to `vint_half_sum` is now a reference.
  a. How did the declaration of `vint_half_sum` change in order to pass the argument by reference instead of by value?

     It doesn't name a new vector, it uses the reference operator (&) to pass by reference.

  b. How did the definition of `vint_half_sum` change in order to pass the argument by reference instead of by value?

It doesn't use the argument vector<int> v, it uses vector<int>& v such that v is created as a reference to the argument used in calling the function.

   c. With this in mind, understand that you can view the initialization of the formal parameter `vector<int>& v` in the function call to `vint_half_sum` on line 34 with `vint` as:

```
vector<int>& v = vint;
```

- Compile and run the source code.
- Carefully, observe the output printed to the screen and then answer the following questions in the fields provided:
   a. Explain why the modification of `v` in `vint_half_sum` *does* mutate the actual argument `vint:`

   Modifying v does mutate vint because v is a reference to vint, meaning that their memory addresses are the same and any modification to v or vint will change the other vector as well.

   b. What information included in the output produced by the calls to `vis::print` in `main` with `vint` and in `vint_half_sum` supports your response to 8a?

   The memory addresses are all the same for each index, and the contents of v and vint are the same before and after the call to vint_half_sum.

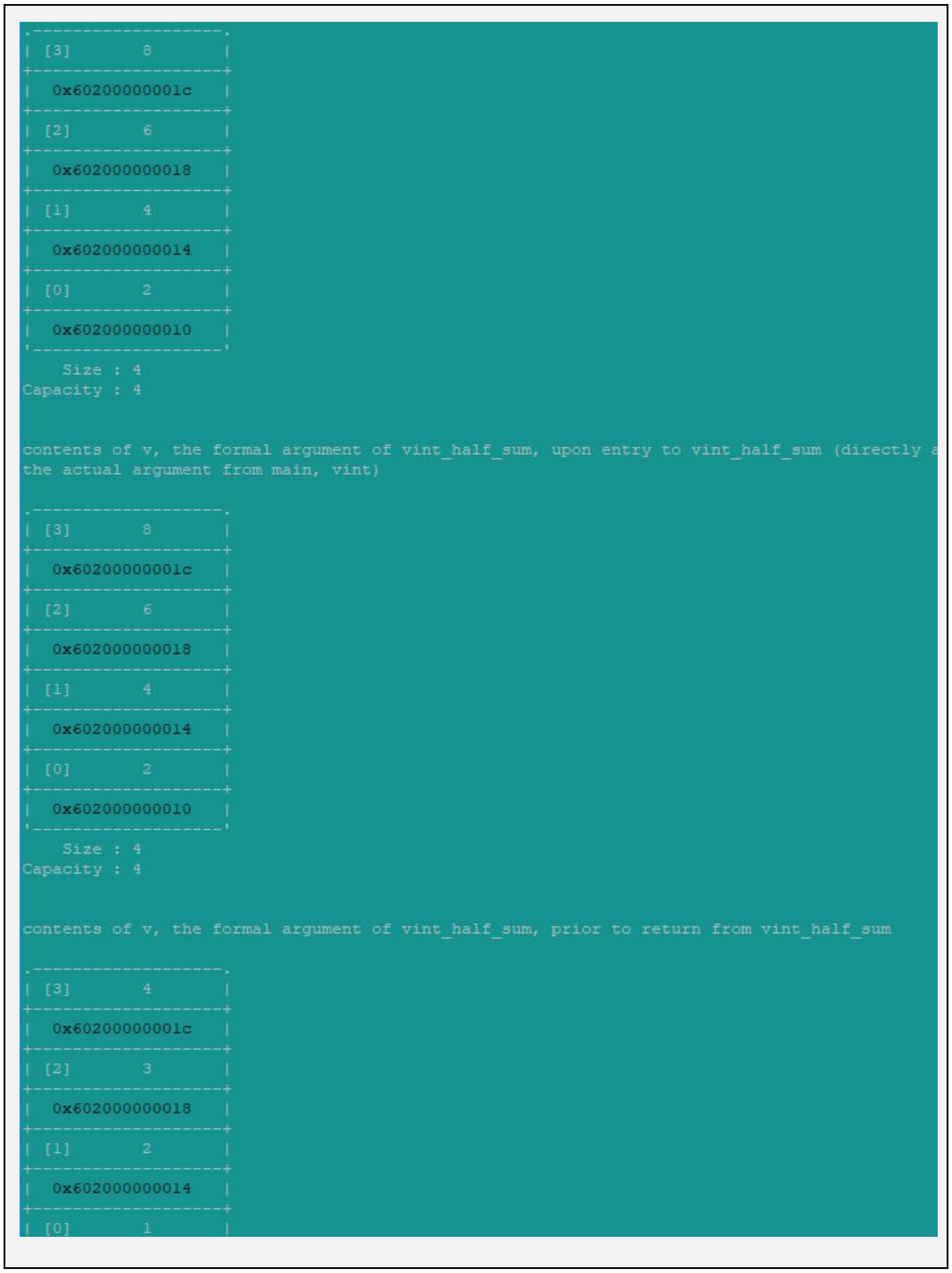
   c. What is the difference between pass-by-value and pass-by-reference? When might you use one over the other? Do you see any potential pitfalls in using either method?

Passing-by-value does not mutate the original input, but rather creates a new variable that holds that same value.
Pass-by-reference does mutate the original input while also creating a reference variable that holds the same value. You might use pass by reference when you want to calculate some computation of the vector vint after calculating the “half-sum” of v. A potential pitfall in using either is keeping track of the variables and making sure you’re using the correct variable name and value, such that there’s no confusion. Having multiple names for the same variable is always confusing.

d. It is recommended that when passing a parameter by reference to a function you should denote in the function’s identifier if it modifies the object in which a formal argument refers. Why do you think that this is a good idea? For instance, why might we decide to update the name of `vint_half_sum` to `half_elems_of_vint_ret_sum`?

Because you may have two functions that compute the same value, but one mutates the argument and one does not. It would cause future errors if you called the wrong function and ended up modifying the original argument when you didn’t mean to, and vice versa.

e. Capture the output written to the terminal window by this program in the form of a screenshot; if you cannot include everything, that’s okay. Drag and drop or paste your screenshot into the box below:



```
.-------------------.
| [3]        8      |
+-------------------+
|  0x60200000001c   |
+-------------------+
| [2]        6      |
+-------------------+
|  0x602000000018   |
+-------------------+
| [1]        4      |
+-------------------+
|  0x602000000014   |
+-------------------+
| [0]        2      |
+-------------------+
|  0x602000000010   |
'-------------------'
    Size : 4
Capacity : 4


contents of v, the formal argument of vint_half_sum, upon entry to vint_half_sum (directly a
the actual argument from main, vint)

.-------------------.
| [3]        8      |
+-------------------+
|  0x60200000001c   |
+-------------------+
| [2]        6      |
+-------------------+
|  0x602000000018   |
+-------------------+
| [1]        4      |
+-------------------+
|  0x602000000014   |
+-------------------+
| [0]        2      |
+-------------------+
|  0x602000000010   |
'-------------------'
    Size : 4
Capacity : 4


contents of v, the formal argument of vint_half_sum, prior to return from vint_half_sum

.-------------------.
| [3]        4      |
+-------------------+
|  0x60200000001c   |
+-------------------+
| [2]        3      |
+-------------------+
|  0x602000000018   |
+-------------------+
| [1]        2      |
+-------------------+
|  0x602000000014   |
+-------------------+
| [0]        1      |
```

Submission

- Save this completed labwork as a PDF [File -> Download As -> PDF Document (.pdf)] and submit to Gradescope for grading.