Activity 8

Comp 11 - Summer Session — Pointers

With a partner(or two), discuss the following code sample and answer the questions below. The instructor and teaching assistants will let you discuss and then be around to answer questions. [1]

8.1 Description

For this activity we will do a few pointer exercises. When working with pointers, it is useful to draw pictures.

8.2 Questions

1. What does the ampersand (&) operator mean in the context of what we covered? How does it work?

2. What is the difference between pass by value and pass by reference when referring to functions?

[1] Activities do not need to be returned to instructors, they are for your benefit.

3. What do you think the terms function caller and function callee mean? See if you can fill out the example below?

```
#include <iostream>

//        Function baz is a callee of?
void baz(){   // ----------------------------
  std::cout << "hi";
}

//        Function bar is a callee of?
void bar(){     // ----------------------------
  baz();
}

//        Function foo is a callee of?
void foo(){   // ----------------------------
  bar();
}


int main(){

  //      Function foo is a callee of?
    foo();       // ----------------------------

  return 0;
}
```

Listing 8.1: Terminology Review

4. What does it mean to dereference? What is the operator(i.e. syntax) that we use to do so?

5. Lets say I have the following example with an array. If I tell you that this is correct code, what do you think is going on?

```
#include <iostream>

int main(){

    int array[5];

    array[0] = 2;
    array[1] = 4;
    array[2] = 6;
    array[3] = 8;
    array[4] = 10;

  // Now why does this work?
    for(int i =0; i < 5; ++i){
        std::cout << *(array+i) << "\n";
    }

    return 0;
}
```

Listing 8.2: Pointer Example 1

```cpp
#include <iostream>

int main(){

    int p = 5;     // Create a new integer
    int* pk = &p; // Create an int pointer.
            // This int pointer points to the
                    // address of p.
    // ======== Now what does this code doe? ========
    *pk = 32;
    std::cout << "p is: " << p << std::endl;
    // (1) What does this print out?

    return 0;
}
```

Listing 8.3: Pointer Example 2

```cpp
#include <iostream>

void increment(int &r){
  r++;
}

int main(){

    int a = 7;
    std::cout << "a is: " << a << "\n";
    increment(a);
    // (2) What is the value of a?
    std::cout << "a is: " << a << "\n";
  // a is ------------

  return 0;
}
```

Listing 8.4: Pointer Example 3