CS 162 Worksheet 3

1. Understand terms:

● Reference – Essentially an alternative name that refers to an existing variable (but implicitly works like a pointer under the hood)

● Pointer – a variable that stores the memory address of another variable / value

● Dereference – access the value in memory location pointed to by a pointer (AKA indirection)

● Statically allocated memory – memory whose sizing, layout, and lifetime is determined during compile time, lives on the stack.

● Dynamically allocated memory – memory whose sizing, layout, and lifetime is determined at runtime, lives on heap.

● Allocate – create memory (or rather, dedicate some memory for some particular data). In C++, use “new” to allocate dynamic memory during runtime.

● Deallocate / free – delete memory (or rather, mark memory as “no longer in use” so that it can be recycled for other data). In C++, use “delete” keyword to free dynamic memory during runtime.

2. Reference vs. pointer syntax

Ex. For a – f, state whether the \* is 1) declaring a pointer or 2) dereferencing a pointer:

1. char\* p;
   1. declaring a pointer
2. x = \*p + 1;
   1. dereferencing
3. int\* ptr;
   1. declaring
4. \*ptr = 5;
   1. dereferencing
5. (\*ptr)++;
   1. dereferencing
6. char\* p1[10];
   1. declaring

Ex. For a – d , state whether the & is 1) declaring a reference variable or 2) address-of operator:

1. string &var2 = var1;
   1. reference varaible
2. void func (double& num1);
   1. reference variable
3. int\* ptr = &value;
   1. address-of operator
4. int\*\* ptr2 = &ptr;
   1. Address-of operator

3. Consider the following declarations:

int x = 100, data[3] = {1, 2, 3};

int \*pi = &x;

int \*\*ppi = &pi;

int &ref = x;

For each of the following expressions, indicate the data type.

Possible options: int, int\*, int\*\*, int\*\*\*

a. x int c. ref int

b. &x int\* d. &ref int\*

e. pi int\* h. data int\*

f. \*pi int i. &data int\*

g. &pi int\*\* j. data[0] int

k. ppi int\*\* m. \*ppi int\*

l. &ppi int\*\*\* n. \*\*ppi int

Common mistake: What’s wrong with the following code?

int\* area (int width, int height) {

int answer = width \* height;

return &answer;

}

int main () {

int w = 5, h = 10;

int \*a = area(w, h);

cout << \*a << endl;

return 0;

}

Out of scope

Takeaway: Never return a pointer (or reference) to a local variable!

4. Dynamic Memory

Ex. Fill in the blanks: Free/delete the memory:

a. \_\_\_\_int\*\_\_\_\_\_\_ data = new int; \_\_\_\_\_\_\_\_delete data;\_\_\_\_\_\_\_\_\_\_;

b. \_\_\_\_\_char\*\_\_\_\_\_ data = new char; \_\_\_\_\_\_\_\_\_\_delete data;\_\_\_\_\_\_\_\_;

c. \_\_\_\_\_char\*\_\_\_\_\_ data = new char[100]; \_\_\_\_\_\_\_\_delete[] data\_\_\_\_\_\_\_\_\_\_;

d. \_\_\_\_\_\_double\*\_\_\_\_ data = new double[20]; \_\_\_\_\_delete[] data\_\_\_\_\_\_\_\_\_\_\_\_\_;

e. \_\_\_\_string\*\_\_\_\_\_\_ data = new string; \_\_\_\_\_\_\_\_delete data\_\_\_\_\_\_\_\_\_\_;

f. \_\_\_\_char\*\*\_\_\_\_\_\_ data = new char\*[10]; \_\_\_\_\_\_\_\_\_delete[] data\_\_\_\_\_\_\_\_\_;

5. C++ strings

In C++, the string type (a class type) is provided to handle strings of text. We didn’t have much time to cover C++ strings in lecture, so you’ll learn about them now. To use them, include the <string> library. Answer the following questions regarding C++ strings:

1. How do you declare a string variable named my\_str and initialize it with the text, “hello world”?

#include <string>

Std::string my\_str = “hello world”;

b. How do you print the contents of my\_str to the terminal?

Std::cout << mystr << std::endl;

1. How do you access a single character in my\_str? For example, how do you change the first character to ‘H’? (Hint: In this regard, a C++ string behaves just like an array of characters).

my\_str[0] = ‘H’;

1. (You may or may not know the answer to this question, depending on how much time we had to cover strings in lecture) How do you get the length (size) of my\_str? Related question: How do you access the very last character of my\_str?

My\_str.length(): or my\_str.size(); my\_str[my\_str.length() - 1];