CCNU-UOW
CSCI851 Advanced Programming
Autumn 2020

Prof. Zhifeng Wang

Laboratory Exercise 3 (Week 3)

1 Warm–up exercises

1. `Thinking:` What is the point of the debugging exercises? Most of the exercises include problems relating to some recently covered material. It's not enough that the program compiles, you can just comment out most of the code if that's the only measure. If your program compiles, see if it actually does what it is supposed to do and think about what it is or isn't doing.

2. Debug: `Debug-A.cpp`. This is based on an exercise from:

   `Joyce Farrell, Object Oriented Programming Using C++, 3rd Edition, Thomson Learning.`

   Generally with debugging you should be making sure it not only compiles but also performs sensibly.

3. Debug the code in the compressed file `Debug-B.zip`. To extract (unzip) the files in `Debug-B.zip` you should use:

   ```
   $ unzip Debug-B.zip
   ```

4. Basic problems with some of the recently introduced concepts. Explain what the problem is with each of these ...

   (a) Qualifier: `const`.

   ```
   const int value = 128;
   value = 256;
   ```

   (b) References:

   ```
   int &reference;
   ```

   (c) Array declaration, this differs between CC and g++.

   ```
   int post = 4;
   int postCode[post];
   ```

   (d) Void pointer one: How do you point the void pointer somewhere?

   ```
   void *vptr;
   cout << vptr << endl;
   ```

   (e) Void pointer two: How do you output the content the void pointer points to?

   ```
   int i;
   int *ptr = &i;
   void* vptr= ptr;
   cout << vptr << endl;
   ```

2 Task Two: Type sizes ...

1. Run through the basic types and determine their sizes.
2. What is the `sizeof` an array? Try something like these ...

```
char A[]="Elephant";
string B="Elephant";
int C[8]={5};
```

How do these differ?

3 Task Three: Arrays and pointers

Write two C++ programs to do the following:

1. Declare an array of three integers.
2. Write a loop that accepts three values from `standard in`.
3. Write a different loop that displays the three values on `standard out`.
4. In the first version, `t1a.cpp`, you are to use subscripts.
5. In the second version, `t1b.cpp`, you are to use pointers only.

4 Task Four: Multiple files, functions and reference variables

Modify Task Four of the Week 2 lab so it works with pass by reference rather than passing by value. If you didn't complete that exercise, choose two or three of the functions in that and implement those across multiple files using pass by value, and then convert the resulting code to pass by reference. Ideally you are keeping copies of these different versions.

5 Task Five: Handling main arguments

See slides 19 and 20 of S2b.

1. Write a program with the more general main function, so with arguments. Use those arguments in a somewhat useful way.
2. Explore the use of the family of argument handling functions:

```
atoi : argument to int
atol : argument to long
atoll : argument to long long
```

6 Task Six: Range `for` and `auto`

These are both useful C++11 functionality. When you compile you wil need to use something like

```
$ CC -std=c++11 file.cpp
```

First the range `for`, which takes care of worrying about the bounds of a for loop. Try the following ...

```
string str("This is a string");
for (char c : str)
cout << c << endl;
```

We can go further by replacing `char` with `auto`, which then also handles the type of the elements. See the impact of using `auto`.

Write a Boolean function to determine if c is a vowel. Use this function so that with outputting the string we put an asterisk instead of a vowel. Boolean functions have a return type of `bool`.

You can modify the program to handle a string entered by the user. Use

```
getline(cin,str);
```

Modify this so rather than a line it takes input until a full stop is entered. This allows you to deal with multiple lines at once.

7 Task Seven: A shortcut to vectors

This task is just to play around with vectors a bit. Make sure you know how to output the elements.

```
size_t size;
cout << "Enter the size of the container: ";
cin >> size;

vector<int> intArray( size );

for(int i=0; i<size; ++i)
   intArray[i] = i;
```

8 * Task Eight: Function pointers

Write a function and set up a function pointer that points to it. Write another function that can make use of your first function, and can accept the function pointer, and illustrate how it does make use of the function pointer.