CS 280
Programming Language Concepts

Spring 2025

C++ Basics

C++ Basics

- All subprograms are referred to as functions.
- A C++ program is a collection of one or more functions.
- Every C++ program must have a function called main
  - The main function is where the program starts
  - When main is finished, it returns control to the operating system and the program is done
- **functions**
  - To define a function you must specify
    - function name
    - type for the value that that the function returns when it is done
    - Parameters (name and type) that are passed as arguments to the function
    - the body of code that is executed when the function is called

C++ Basics

```
//============================================================
// Name        : HelloWorld.cpp
// Description : Hello World in C++
//============================================================
// stuff that begins with // is a comment

/* this is also a comment */
#include <iostream>
using namespace std;

int main(int argc, char *argv[]) {
        cout << "Hello, World!" << endl;
        return 0;
}
```

C++ Basics

```
// the code below says
// main is the name of the function
// the type that it returns is int
// main takes two arguments named argc and argv
// the two statements inside of the { } are run when
// main gets called
// A return of 0 value indicates that everything
// went with no problems

int main(int argc, char *argv[]) {
   cout << "Hello, World!" << endl;
   return 0;
}
```

C++ Basics

- Java programmers: notice that there is no “class” keyword in this program.
- In C++, everything does NOT have to be in a class.
- We don’t need a class for this simple example. We will just write a main function.
- Functions do NOT have to be methods defined inside a class.

Preprocessor

- Preprocessor macros (instructions) are commonly used to specify that code from another file is to be included
- A preprocessor processes a program immediately before the program is compiled to expand embedded preprocessor macros
- A well-known example: C/C++ preprocessor
  - expands `#include, #define`, and similar macros
  - It tells the compiler to include the contents of a file into the body of the code before compiling; similar in concept to "import" in other languages.
  - A filename inside of < and > means that the file is in the standard place for the standard files that come with the compiler.
  - You can make your own include files and #include them, enclosing filenames in " and "
- By convention, these included files are called "header files" and usually have names ending in .h

Header or Include Files

- C and C++ rely on including files in the preprocessor phase of the compile to make sure that all programs have common definitions of things: variables, constants, functions declarations and different data types.

- Header files are often associated with library implementations.
- C++ comes with a lot of standard libraries and header files.

using namespace

- C++ can put functions and definitions into a separate, labeled *namespace*
  - Java programmers: this is a little like packages
- To access something in a namespace, you need to preface the item you are accessing with the name of its namespace and :: (scope resolution operator)
- Items in the C++ standard library are in a namespace named “std”, so cout is actually std::cout

```
namespace std{
   . . . .
   //Includes declarations of variables, data types, etc
   . . .  .
}
```

- Saying “using namespace std;” tells the compiler that you want to have everything in the std namespace visible in your program
- You could also indicate that you’re just using one symbol: “using std::cout;”

Object Files

- Each Object File contains:
  - executable code
  - global variables
  - constants
  - “linkage information”
    - what code and data do I provide that others may use
    - what code and data do I require that I need others to provide for me
- A library is just a collection of object files (sometimes an “archive” or a “.a” or a “library” or a “.lib”)

iostream

- iostream is a C++ standard library that provides definitions and implementations for input streams (istream) and output streams (ostream).
- At runtime OS provides every program with:
  - “standard input” or “standard in” (in Java, System.in)
  - “standard output” or “standard out” (in Java, System.out)
  - “standard error” or “standard err” (in Java, System.err)

- To read what a user types in, read from the standard *in*
- To write something for the user to see, write to standard *out*

iostream

- The iostream library provides a type-safe way to access standard *in* and standard *out*.
  - cin object is an input stream connected to standard in
  - cout object is an output stream connected to standard out
  - cerr object is an output stream connected to standard err

- iostream uses the << operator to write to a stream (Insertion Operator)
- iostream uses the >> operator to read from a stream (Extraction Operator)
- There are also methods defined for various operations on streams

iostream

```
//the line below performs the << operation on cout:
// stream << something
//causes the something to be written to the stream.
//In this case, the something is a string of characters.
//
//<< endl causes an end of line to be written to the
//stream. If you leave off endl you will not skip to the
//next line. Writing "\n" or '\n' does the same thing.

cout << "Hello, World!" << endl;
```

iostream

- The << operator is defined to “shift information out to the stream”
  - This code writes out an integer:

```
int x;
cout << x;
```

- The >> operator is defined to “shift information in from the stream”
  - This code reads in an integer into an integer type variable:

```
int x;
cin >> x;
```

  - This code reads in a string into a string type variable. Note that string is not a built-in type. It is a class which requires to be included in your source code as:

```
#include <string>
. . .
string name;
cin >> name;
```

Strings Overview

- **`string`** features
  - Not necessarily **`null`** terminated
  - **`length`** member function: **`s1.length()`**
  - Use **`[]`** to access individual characters: **`s1[0]`**
    - **`0`** to **`length-1`**
- Assignment
  - **`s2 = s1;`**
    - Makes a separate copy
- Concatenation
  - **`s3.append( "pet" );`**
  - **`s3 += "pet";`**
- Comparing strings
  - Overloaded operators: **`==`**, **`!=`**, **`<`**, **`>`**, **`<=`** and **`>=`**

Strings Overview

- Range checking
  - **`s3.at( index );`**
    - Returns character at **`index`**
    - Can throw **`out_of_range`** exception
  - **`[]`** has no range checking
- Substring
  - **`myStr = "Concepts of Programming Languages";`**
  - **`str = myStr.substr(12, 7);`**
- **`s1.find( s2 )`** Searches a string for the first occurrence of a particular substring:
  - If found, index returned. If not found, **`string::npos`** returned
  - **`s2 could be a literal string, or a string expression`**

Strings Overview

- Conversion functions to C-Style char * Strings
  - **`string`**s are not necessarily null-terminated.
  - **`s1.copy( ptr, N, index )`**
    - Copies **`N`** characters into the array **`ptr`**
    - Starts at location **`index`**
    - Need to null terminate
  - **`s1.c_str()`**
    - Returns **`const char * (constant char pointer)`**
    - Null terminated

Strings Overview: Testing Digits Example

```
#include <iostream>
#include <string>
#include <cctype>
//isdigit example
using namespace std;
int main(){
   string str = "hds24w7ds", num = "";
   //instead use getline to read a line from keyboard
   int value, i = 0;
   while (i < str.length()){
     if (isdigit(str[i]))
      num += str[i]; i++;
   }
   cout << "The digit characters form the number: "<< num;
   return 0;
}
```

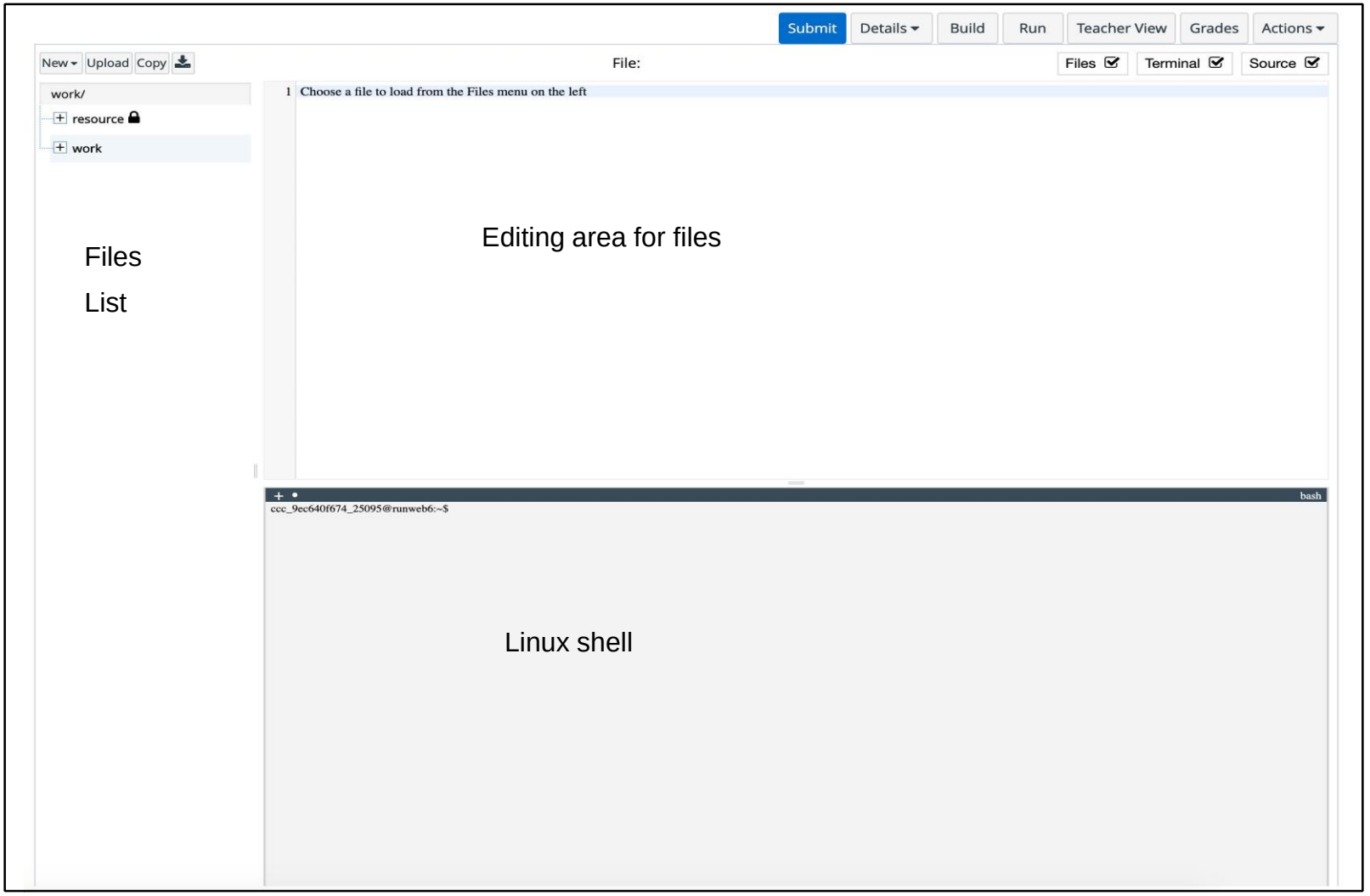
Notes on Programming Assignments and Projects

- All programming assignments must be written in C++-11.
- You will NOT need to submit a printout of your code.
- All programming projects must be submitted electronically as a *.cpp or *.h files to Vocareum through Canvas.
- **Every file should include a block of identifying comments at the top of each file**.
- **Please note that emailed submissions will NOT be accepted**.

Vocareum Environment

- Vocareum is a cloud-based System supporting course work in many areas, including computer science.
- Vocareum system can be linked via Canvas.
- Vocareum has features to automate the building, running, submitting and grading of your assignments.
- Vocareum has a Linux terminal window in addition to a window to edit your code.
  - You may edit your code in Vocareum OR you may upload code from elsewhere **(recommended)**. However, please note that **The code submitted to Vocareum is what will be graded.**
  - Vocareum has buttons to allow you to build, run, and submit your code.
  - You may press the submit button for your assignment as often as you like. Only the last submission will be considered as your grade. The time that you press the submit button is the time recorded as your submitted work.

Vocareum Environment



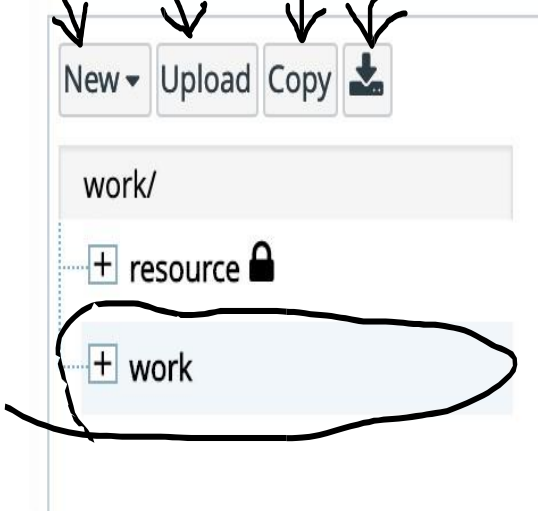
Vocareum Environment

Select to create new files

Click to upload

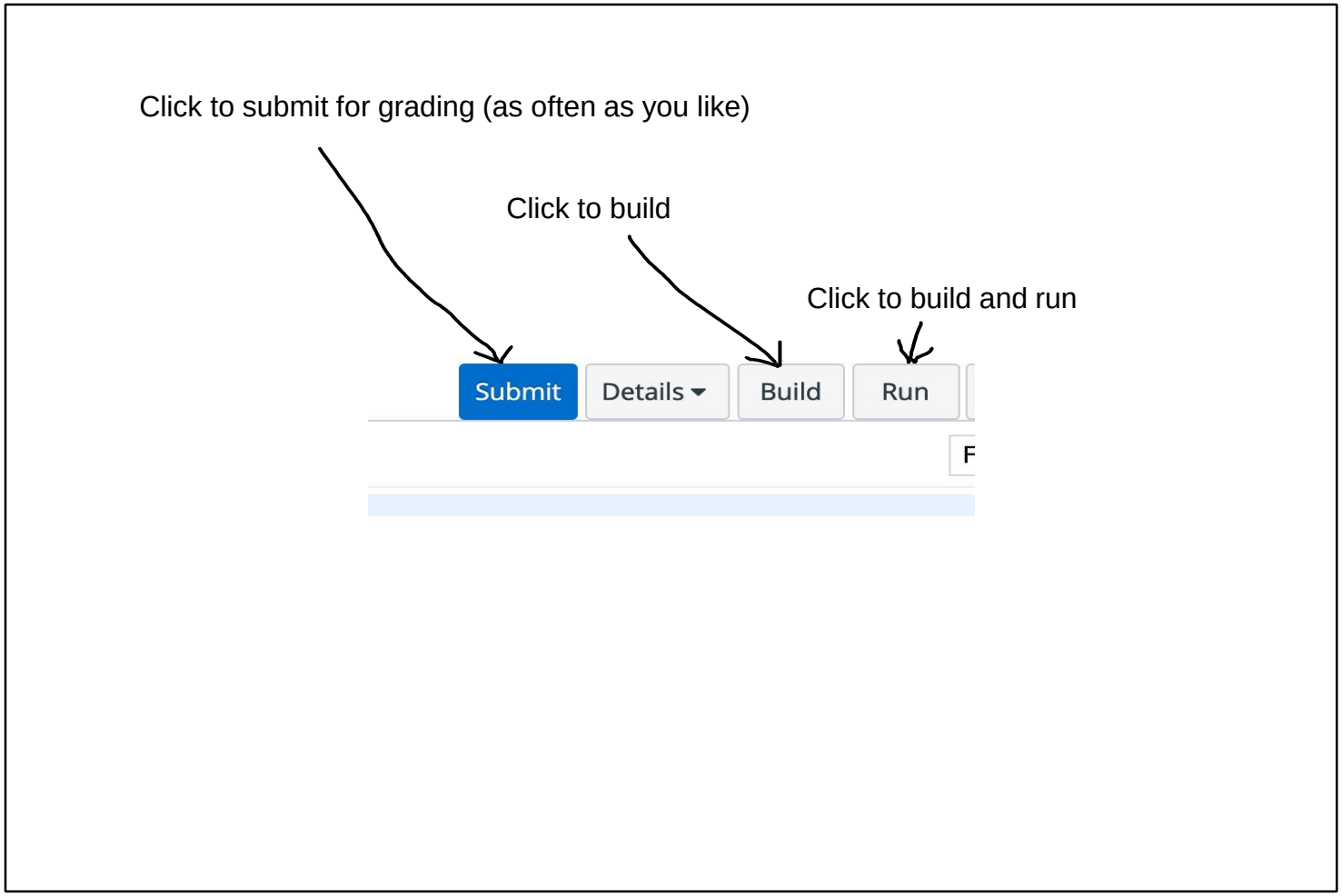
Click to copy

Click to download



Work folder selected

Vocareum Environment



Vocareum Environment

- If your program does not compile, it will be as if you did not do the assignment.
- Partial answers or incorrect output will reduce your grade.
- Vocareum has cheat detection software built into it. Be aware that I may use it.
- Each assignment in Vocareum has a “Work” folder. When you “Submit” your assignment, you are submitting your entire work folder.
- The system will assume every C++ source file in the “Work” folder is part of your assignment, and will compile everything and link everything together.
- DO NOT #include a .cpp file to add it to your program
  - For projects, the system will automatically compile every .cpp file and link them together.

Integrated Development Environments

- There are many IDEs available
  - Xcode comes with Macs
  - Microsoft Visual Studio is available on Windows
  - Eclipse is a free to download
  - Dev C++ is free to download (recommended)

- If you are comfortable with one of these, pick it and use it; just make sure that you upload your code to Canvas/Vocareum.
- Make sure you have a C++ compiler that supports C++11.

Using Linux

- Download the latest gnu C++ compiler, or install it using whatever package management system you have on your machine.
- Use the most recent version available
- The compiler is called g++
- Make sure you compile using the command line option
  - –std=c++11 to activate C++11 features. For example, use the command:

    g++ --std=c++11 –g myprog.cpp –o myprog

Using a Macintosh

- The clang compiler and Xcode are excellent tools.
- There is a checkbox in preferences to activate C++ 11 features; make sure that you check it.

Using Windows - VS

- Visual Studio is available for free download for students.
- Be aware that there are several non-portable windows-only things that will prevent your code from working on Vocareum.
- If your code works on VS but does not work on Linux or Vocareum, then it is “not portable” and from the perspective of the course, it does not work.

Using Windows – Not VS

- Windows users can get the GNU C++ compiler for free from Cygwin or mingw or msys2
  - You can use the command line or you can use it with an IDE.
  - MSYS2 installation: See the following link for instructions
    - https://www.msys2.org/wiki/MSYS2-installation/
  - Cygwin installation: See the following link for instructions
    - https://www.cygwin.com/install.html
  - MinGW installation: See the following link for instructions
    - https://sourceforge.net/projects/mingw/

Short Assignment 1: Welcome Message

- *The objective of this simple assignment is to make sure you can edit, compile and run a C++ program on your IDE, and then upload the source code for automatic grading by the Vocareum system.*

- Write a C++ program that displays a message as "CS 280 (Spring 2025)", followed by a prompt for a student to enter his or her first name, last name and section number, respectively. The program should read the student's first and last names as strings, and the section number as an integer, then display a welcoming message. A full dialogue example is as follows:

```
Welcome to CS 280 (Spring 2025)
What are your first name, last name, and section number?
George Fritz 4
Hello George, Welcome to CS 280 Section 4.
```

Short Assignment 1: Welcome Message

- **Note: Use the same format and phrases for the input and output as shown for full credit. However, testing your program on your environment would require an input from the user to be entered for the names as shown in italic lines. For automatic grading on Vocareum, the input will be provided from an input file instead.**
  - **The correct output file on Vocareum is "input.correct" as shown below.**
  - **The correct output shows what you would have generated, excluding what a user have entered as an input.**
    - **Automatic grading will relax matching outputs based on case sensitivity.**

```
Welcome to CS 280 (Spring 2025)
What are your first name, last name, and section number?
Hello George, Welcome to CS 280 Section 4.
```