601.220 Intermediate Programming

Summer 2024, Meeting 13 (July 1th)

Today's agenda

$\longrightarrow$ Have a physical id.

- Midterm exam Wednesday July 3th.
- Day 22 recap questions
- No exercise today!
   - There will be time for additional questions before midterm on Wednesday.

Reminders/Announcements

- HW5 is due **Thursday, July 11th**. Homework will be released this Thursday.
  - Note that it depends on some material that will be covered in class on Friday and Monday
  - We will have covered almost everything you will need to know by the end of class on Friday

Day 22 recap questions

1. What is the difference between C and C++?
2. What is a namespace in C++?
3. Why should you not put "using" statements in header files?
4. How do you read and write in C++ (i.e. standard inputting/outputting)?
5. What is the difference between C strings and C++ strings?
6. How long can a C++ string be?

1. What is the difference between C and C++?
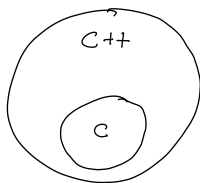
C: "portable assembly language"

Very little direct support for rich data types.

Very little support for automatic resource management.

No support for generic programming.

Almost no run-time support is needed.

C is a relatively simple language.

C vs. C++

C++: a modern object-oriented language

$\longrightarrow$ String

C++ classes allow the creation of very rich data types.

Constructors and destructors shift the burden of resource management (memory, files, etc.) onto the compiler.
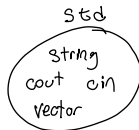
Extensive support for generic programming (template classes and functions.)

Some runtime support (e.g., exceptions) is required.

C++ is arguably the most complex programming language ever created.

C11 , C14, C17 , C20

2. What is a namespace in C++?

std

- string
- cout cin
- vector

my – custom

- string
- vector

A namespace allows names of classes, functions, data types, etc. to be isolated so that they don't conflict with other classes/functions/data types that happen to have the same name.

E.g., all C++ standard library classes are in the std namespace. For example, std::string is the string class provided by the standard library. You could define your own class called "string", and it would not conflict with std::string.

If you are developing a library that you intend to be incorporated into other programs, put all of its classes/functions/data types/etc. into a library-specific namespace.
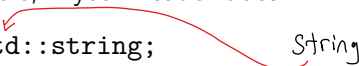
std::string
my_custom::string

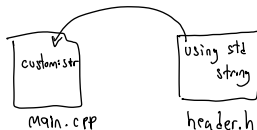# 3. Why should you not put "using" statements in header files?

Because you would be forcing any code that #includes the header file to accept the name(s) imported by the using statement.

For example, if your header does

using std::string;   *String*

then you make it difficult for code #includeing your header to use any other class called "string".

4. How do you read and write in C++ (i.e. standard inputting/outputting)? $C \Rightarrow$ printf , scanf (functions)

Writing data: use std::cout. Reading data: use std::cin. You need to #include <iostream> to use these.

Example:

```cpp
std::string name;
int age;

std::cout << "What's your name? ";
std::cin >> name;

std::cout << "How old are you? ";
std::cin >> age;

std::cout << "Hello, " << name << ", nice to meet you\n";
```

5. What is the difference between C strings and C++ strings?

Char str [ ] = "Hello world";

C strings:

Stored in an array of `char` elements.

Entirely the programmer's responsibility to ensure the string is not too large for the array.

Dealing with arbitrary-length strings (e.g., reading lines of text from a file) is quite tricky, and will generally involve dynamic allocation and resizing buffers on the fly.

C strings vs. C++ strings

C++ strings:

Storage for the character sequence is handled automatically.

Storage grows as needed! And, is automatically freed when the string is destroyed.

Dealing with arbitrary-length strings is easy.

6. How long can a C++ string be?

As long as it needs to be, limited only be available memory.

Notes

* link list

* Bit wise operations

* multi dimensional arrays

* Memory management. (pointers)

---

* Compilation process
* random number generation
* recursion
* Filer.

* Don't Forget about midterm contribution survey.

Notes

Notes

Notes

Notes