C++: Tour Pt 1

Before We Begin

- CS4 Newsgroup

Plan for today

- Data types
- Variables / Pointers
- Operations
- Statements

Data Types

- Basic data types
  - `int`, `short`, `long`, `unsigned`
  - `bool`
  - `char`
  - `float`
  - `double`
  - `void` (sort of) – more on `void` later
- Not classes!
- No corresponding classes like in Java

Data types

- Strings
  - No basic string type in language but…
  - C style strings
    - In C strings are arrays of `char` with the string terminated by a null character

h	e	l	l	o	\0

  - C++ standard template library does include a string class.

Data Types

- The class
  - Like in Java, classes have
    - Data members
    - Methods
  - Constructor
    - called when a new instance is created
  - Destructor
    - Called when an instance is destroyed

Variables in C++

- Variable types
  - Basic
  - Pointer
  - Reference

Variables in C++

- Basic variable
  - Memory associated with a variable with size based on the type of the variable.
  - Variable declarations are "executable" statements
    - Memory is allocated when declaration is made
  - No need to use `new` when instantiating objects of a class.

Variables in C++

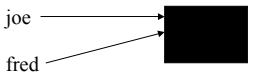
Basic variable

```
int foo;
float f = 7.0;

Student joe ("Geigel" , "Joe", "GCCIS",
  "CS");
```

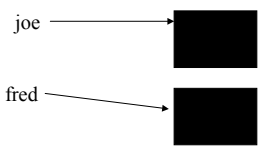
Variables in C++

- Assignment = copy not reference
  - In Java
    - `Student joe = new Student("Geigel" , "Joe", "GCCIS", "CS");`
    - `Student fred = joe;`



Variables in C++

- Assignment = copy not reference
  - In C++
    - `Student joe ("Geigel" , "Joe", "GCCIS", "CS");`
    - `Student fred = joe;`



Variables in C++

- Basic variable
  - Accessing class members
    - Uses the . Syntax used in Java

    - `Student joe ("Geigel" , "Joe", "GCCIS", "CS");`
    - `joe.getGrades();`

Variables in C++

- Pointer Variables
  - Stores the memory address of an object.
  - `new` returns a pointer to an object and allocates memory for it on the heap (free store).
  - Can have pointers to basic data types.
  - C++ has no garbage collection!
  - NULL pointer takes value 0.

Variables in C++

Pointer variable

```
int *foo;
float *f = 7.0; // Invalid
float *g = 0;  // okay
float *h = 0x12345; // actually illegal!!

Student *joe = new Student ("Geigel" ,
  "Joe", "GCCIS", "CS");
```

Variables in C++

- Pointer Variables
  - Dereference operator *
  - If `ptr` is a pointer
    - i.e A variable whose contents is a memory address
  - then `*ptr` refers to the object or data item that is pointed to by ptr
    - Can be interpreted as:
      - The data item or object at `ptr`
      - The object or data item pointed to by `ptr`

Variables in C++

Pointer variable

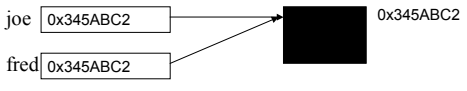
```
float *f = 7.0; // Invalid however
float *f = new float;
(*f) = 7.0;
Student *joe = new Student ("Geigel" ,
  "Joe", "GCCIS", "CS");
Student *fred = joe;
```

Variables in C++

- Address of operator
  - You can always get the address of any variable or object by using the address of operator &.
    - `float f = 7.0;`
    - `float *fptr = &f;`
  - Use with caution!!
    - `Student joe ("Geigel" , "Joe", "GCCIS", "CS");`
    - `Student *joeptr = &joe;`

Variables in C++

- Assignment = copy not reference
  - For Pointer Variables as well
    - `Student *joe = new Student("Geigel" , "Joe", "GCCIS", "CS");`
    - `Student *fred = joe;`



Variables in C++

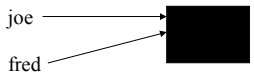
- Pointer variable
  - Accessing class members
    - Uses the -> Syntax
    - `Student *joe = new Student ("Geigel" , "Joe", "GCCIS", "CS");`
    - `joe->getGrades();`
    - Which is the same as
    - `(*joe).getGrades();`

Variables in C++

- Reference Variables
  - Alias for an already existing object
  - Usually used to pass function arguments by reference.
  - Cannot change what they point to

Variables in C++

- Reference Variables
  - `Student joe ("Geigel" , "Joe", "GCCIS", "CS");`
  - `Student &fred = joe;`



Variables in C++

- Reference variable
  - Accessing class members
    - Uses the . Syntax
    - `Student joe ("Geigel" , "Joe", "GCCIS", "CS");`
    - `joe.getGrades();`
    - `Student &fred (joe);`
    - `fred.getGrades();`

Variables in C++ -- Examples

```
Student joe ("Geigel" , "Joe", "GCCIS", "CS");
Student *joe_ptr (new Student ("Schmoe" , "Joe", "GCCIS",
   "CS");
Student &joe_ref (joe);

// okay
joe = *joe_ptr;
*joe_ptr = joe;
joe_ref = *joe_ptr; // changes joe
joe = joe_ref;

// dangerous
joe_ptr = &joe;
joe_ptr = &joe_ref;
```

Variables in C++

- Questions?

Operations

- Numeric
  - Usual arithmetic
    - + , - , * , /, % (mod)
  - Operate and assign
    - +=, -=, *=, /=, %=
  - Increment, Decrement
    - ++, --

Operations

- Logical
  - <. >, <=, >=, ==
    - == is logical equals to
  - &&, ||, !

Operations

- Bitwise operations
  - Can operate on the bits of an integral type
    - Basic logical operations
      - &, |, ^
    - Bit shifting
      - <<, >>
        - » `int a = 0x1111`
        - » `a >> 4;`

Operations

- sizeof
  - Will return the size of a data item or object
    - sizeof (char) = 1
    - sizeof (bool) = 1

Operations

- Questions?

Statements

- `if` (condition) statement
- `if` (condition) statement `else` statement
- `switch` (condition) statement
- `while` (statement) statement
- `do` statement `while` (expression)
- `for` (; ;) statement
  - Statements can be nested blocks of code
    - I.e { … }

Statements

- Logical conditions
  - False if condition evaluates to 0
  - True if condition evaluates to a non-zero value.
  - Can be interpreted as condition != 0

Statements

- Logical conditions
  - `int a;`
  - `if (a) { ... }  // same as`
  - `if (a != 0) { ... }`
  - `int *b;`
  - `if (b) { ... } // same as`
  - `if (b != 0) { ... }`

Statements

- Logical conditions
  - Are short cuircuited
  - `if  ( ( a < b)  && ( c > d)) { …}`
    - If (a > b), (c > d) will not get tested.

Statements

- Logical conditions
  - Assignments and declarations can be made in a logical condition
  - The following is valid:
    - `if ( double d = somefunction (a)) { … }`
  - A most common mistake
    - `if ( a = b ) { … }  // is not the same as`
    - `if ( a == b) { … }`

Statements

- Exiting a loop
  - break – exit the loop
  - continue – perform next iteration of a loop
  - goto – go anywhere

In memory of…

- Edsger Wybe Dijkstra
- 1930 – 2002
- GOTOs Conidered Harmful

Statements

- Finally, my favorite C++ statement
  - var = (condition) ? statement1 : statement2
  - Same as:
    - `if` (condition)
      `var` = expression1
      else `var` = expression2

    - `max = (a <=b) ? a : b;`

Summary

- Part 1 of our tour of C++
  - Data types
  - Variable / Pointers
  - Operators
  - Statement
- Tomorrow:
  - Functions, arrays, basic I/O
- Questions?