CS 101: Computer Programming and Utilization

07- Scratch to C++ basics

Instructor: Sridhar Iyer
IIT Bombay
Activity: Write a program
to convert Fahrenheit to Centigrade

Write it in at least 2 of the following:
• psuedo-code
• Scratch
• C++
• Input F from keyboard
• Set C to 5*(F-32)/9
• Output C to display

```cpp
#include <iostream>
using namespace std;
main() {
    float f, c;
    cin >> f;
    c = 5*(f-32)/9;
    cout << c;
}
```
C++: More detail

main() {
    float fahrenheit;
    cin >> fahrenheit;
    float centigrade = 5*(fahrenheit-32)/9;
    cout << centigrade;
}

- What is a procedure?
- What are data types?
- Where did `cin` and `cout` come from?
- Rules of writing arithmetic expressions
Procedure or function

- Encapsulates a piece of computation and gives it a name
  - E.g. `main` is the default procedure that is run when your program is executed from the shell

- May accept input values stored in named variables
  - E.g. `int max(int a, int b)`

- And return output value
  - E.g. `max(-3, 2)` should return 2
Class discussion:
Why bother with having functions?

Points made by students in both batches:

- Code can be reused
- Ease of testing and debugging
- Modularity is useful for understanding
- Re-implementation is possible
- Abstraction and Encapsulation
Variables - data types

• Computer memory is a 2d array of bits
  • Eight columns (one byte or “B”)
  • Rows depends on how much memory you have; “1 GB” means 1,073,741,824 rows
  • Hard disk is similar, only larger and slower

• What programmers want:
  • Integers, real numbers, complex numbers
  • Characters, strings of characters
  • Arrays, variable length lists, mappings
  • Windows, buttons, menus

• Later we will study how these are represented
Variable declaration

- `float fahrenheit;`
  - Uninitialized, may get garbage on read
- `float fahrenheit = 95;`
- `const float fahrenheit = 9.52e14;`
  - Value will never change
  - Scientific notation saves typing lots of zeros
- `int x = 3, y = x/2;`
  - Can initialize variables based on others already initialized

- Why bother to declare variable names and types?
Why bother to declare

• Variable names
  • What if you type it incorrectly later?
  • To initialize before any use

• Types
  • To check all assignments to the variable
  • To interpret a bit sequence as intended in your program (e.g. float and int are both 32 bits)

• There are languages that do not enforce variable name and type declarations
  • Can be lazy, but generally a Bad Idea
Choosing names

- C++ allows any sequence of characters A—Z, a —z, 0—9, and underscore
- Not starting with a digit
- Up to some maximum number of characters
  - old_style_variable_name
  - newStyleVariableName ("camel case")

- Using single characters for variable names, like 'c' and 'f' as shown on slide 3, is BAD practice!
cin and cout

- “Console in” (keyboard) and “console out” (display)
- These variables are not defined magically
- To use them, must prefix our C++ code with instruction to include a header file like this:
  ```
  #include <iostream>
  ```
- The operating system and compiler work together to let your code access the keyboard and display through cin and cout
- Not quite…
Namespaces

• We must write `std::cin` and `std::cout`

• Two different Ravi Vermas in hostels 2 and 5

• To avoid confusion, write as
  `H2::RaviVerma and H5::RaviVerma`

• “Namespace::” lets libraries written by different people avoid variable and function name clashes

• `std` is the “standard” namespace within which C++ predefined variables and functions are provided
The std namespace and using

• Tedious to type std:: in front of everything

• If you are not using too many namespaces simultaneously, you can choose a default by saying

  using namespace std;

before using things defined inside std.
Compiling your source code

```cpp
#include <iostream>
using namespace std;
main() {
    float fahrenheit;
    cin >> fahrenheit;
    float centigrade = 5*(fahrenheit-32)/9;
    cout << centigrade;
}
```

- At your shell, type
  
g++ cf.cc
- Now run resulting file as
  
  ./a.out

Save to file “convert.cc”

Executable a.out

g++ compiler
Program files and executables

- convert.cc and a.out are files
- A file is a sequence of bytes
- These bytes can be interpreted differently depending on the applications that read or write the files
  - convert.cc is a text file to be written by a programmer and read by the C++ compiler
  - Any name ending in .cc or .cpp is ok
  - a.out is an executable file to be run from the shell command line
  - You can rename this file as you wish – Geany does this automatically for you.
Compilation and execution summary

- Source code: convert.cc
- Compiler: g++
- Main function: main() in a.out
- C/C++ environment: char, short, int, float, double, if, switch, while, ...
- libraries: string, math.h, iostream
- Operating system: Windows, Linux, Mac OS, ...
- Tools: Bash shell, Precompiled libraries, Header files
Think-Pair-Share: Write a program

A petroleum company has erected a number of cylindrical tanks on a rectangular field. External surfaces of these are to be painted, including the flat circular cover on top.

Write a program that, given appropriate inputs, will output the cost of painting.

Think: Identify variables; Write pseudo-code

Pair: Check each others' pseudo-code, converge on one answer and convert into C++

Share: Check with next slide demo06-painting.cpp
```cpp
#include <iostream>
using namespace std;

int main() {
    float r, h, price, cost, pi = 3.14159;
    int Ntanks;

    cout << "give the radius and height of cylinder: ";
    cin >> r >> h;
    cout << "give number of tanks: "; cin >> Ntanks;
    cout << "give price per sq meter for painting: "; cin >> price;
    cost = price * Ntanks * (2 * pi * r * h + pi * r * r);
    cout << "Cost of painting is: " << cost;
    return 0;
}
```

Compilation finished successfully.
What next?

• Next class: Statements, Conditions, Loops, ...
  • You are already familiar with these concepts in Scratch, so you only need to learn C++ syntax!
  • We will move quickly onto advanced constructs

• Next lab: C++ programming
  • Using Geany (Development Environment)
  • Completing your Scratch projects