CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

< □ > < □ > < □ > < □ > < □ >

CSci 127 (Hunter)

Lecture 12

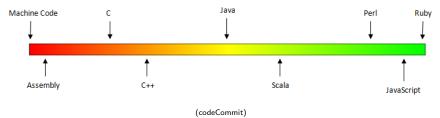
3 1 May 2018 1 / 24

Today's Topics



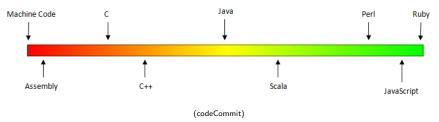
- Recap of Low-Level Programming
- Introducing C++
- Hello, World in C++
- I/O and Definite Loops in C++
- Final Exam Overview

590



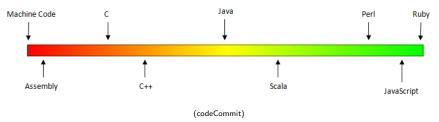
• Can view programming languages on a continuum.

E 990

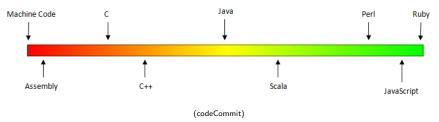


- Can view programming languages on a continuum.
- Those that directly access machine instructions & memory and have little abstraction are low-level languages

Sac



- Can view programming languages on a continuum.
- Those that directly access machine instructions & memory and have little abstraction are **low-level languages** (e.g. machine language, assembly language).

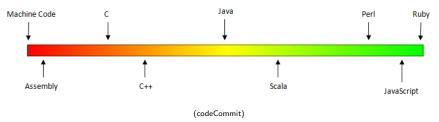


- Can view programming languages on a continuum.
- Those that directly access machine instructions & memory and have little abstraction are **low-level languages** (e.g. machine language, assembly language).
- Those that have strong abstraction (allow programming paradigms independent of the machine details, such as complex variables, functions and looping that do not translate directly into machine code) are called **high-level languages**.

CSci 127 (Hunter)

Lecture 12

1 May 2018 3 / 24



- Can view programming languages on a continuum.
- Those that directly access machine instructions & memory and have little abstraction are **low-level languages** (e.g. machine language, assembly language).
- Those that have strong abstraction (allow programming paradigms independent of the machine details, such as complex variables, functions and looping that do not translate directly into machine code) are called **high-level languages**.
- Some languages, like C, are in between- allowing both low level access and high level data structures. ◆□▶ ◆□▶ ◆三▶ ◆三▶ ○○○

CSci 127 (Hunter)

Lecture 12

1 May 2018

3 / 24

HUX 12:01a 23- 1
A 882888 C2 38 AEP #\$38 A 882882 18 CLC
A 6622603 F8 SED
A 882884 A9 34 12 LDA #\$1234 A 882887 59 21 43 ADC #\$1234
A 88288A BF 83 7F 81 STA \$817583
A 86288E 08 CLD
A 882811 88 HRK
A 2012
r
PB PC MUHARAIZC A X Y SP DP 18 ; 00 E012 00110000 0000 0000 0000 0002 CFFF 0000 00
8 2000
BREAK
PB_PC_NUMADIZC_A_X_Y_SP_DP_DB
; 00 2013 00110000 5555 0000 0002 CFFF 0000 00 n 7f03 7f03
2887F83 55 55 88 88 88 88 88 88 88 88 88 88 88
•

(wiki)

• We will be writing programs in a simplified machine language, WeMIPS.

200



(wiki)

- We will be writing programs in a simplified machine language, WeMIPS.
- It is based on a reduced instruction set computer (RISC) design, originally developed by the MIPS Computer Systems.



(wiki)

- We will be writing programs in a simplified machine language, WeMIPS.
- It is based on a reduced instruction set computer (RISC) design, originally developed by the MIPS Computer Systems.
- Due to its small set of commands, processors can be designed to run those commands very efficiently.



⁽wiki)

- We will be writing programs in a simplified machine language, WeMIPS.
- It is based on a reduced instruction set computer (RISC) design, originally developed by the MIPS Computer Systems.
- Due to its small set of commands, processors can be designed to run those commands very efficiently.

イロト イロト イヨト イヨ

• More in future architecture classes....

"Hello World!" in Simplified Machine Language

ne: 3 Go!	Show/Hide Demos							User Guide	Unit Tests Do
	Addition Doubler	Stav Looper	Stack Test	Hello World					
	Code Gen Save Str	ing Interactive	Binary2 Deci	mal Decim	al2 Binary				
	Debug								
# Store 'Hello worl	dl' at the top of	f the stack				Step	Run	 Enable auto switching 	1
ADDI \$sp, \$sp, -13 ADDI \$t0, \$zero, 72	2 # H					s	т	A V Stack	Log
SB \$t0, 0(\$sp) ADDI \$t0, \$zero, 10)1#e								
SB \$t0, 1(\$sp) ADDI \$t0, \$zero, 10	8 # 1						s0:	10	
SB \$t0, 2(\$sp)							s1:	9	
ADDI \$t0, \$zero, 10 SB \$t0, 3(\$sp)	08 # 1						s2:	9	
ADDI \$t0, \$zero, 11	1 # 0						s3:	22	
SB \$t0, 4(\$sp)							s4:	696	
ADDI \$t0, \$zero, 32 SB \$t0, 5(\$sp)	# (space)							976	
ADDI \$t0, \$zero, 11	9 # w						s5:		
SB \$t0, 6(\$sp)							s6:	927	
ADDI \$t0, \$zero, 11 SB \$t0, 7(\$sp)	l1 # 0						s7:	418	
ADDI \$t0, \$zero, 11	4 # r								
SB \$t0, 8(\$sp)									
ADDI \$t0, \$zero, 10 SB \$t0, 9(\$sp)	08 # 1								
ADDI St0, Szero, 10	b # d								
SB \$t0, 10(\$sp)									
ADDI \$t0, \$zero, 33	3 # 1								
SB \$t0, 11(\$sp) ADDI \$t0, \$zero, 0	# (mull)								
SB \$t0, 12(\$sp)	# (null)								
ADDI \$v0, \$zero, 4 ADDI \$a0, \$sp, 0	# 4 is for print	string							
syscall	# print to the								

(WeMIPS)

In Pairs or Triples:

Predict what the code will do:

```
# This is the same as the doubler, except the jumps cause the order
2
  # to change drastically, therefore all of the values will be different.
  CHANGE S: ADDI $t0, $zero, 2
4 BEQ $s0, $t0, EXIT
5 ADD $s1, $s0, $s0 # double s0 by adding it to itself, should be 4
6 ADD $s2, $s1, $s1 # double s1 by adding it to itself, should be 8
  ADD $s3, $s2, $s2 # double s2 by adding it to itself, should be 16
8 ADD $s4, $s3, $s3 # double s3 by adding it to itself, should be 32
9 ADD $s5, $s4, $s4 # double s4 by adding it to itself, should be 64
10 ADD $s6, $s5, $s5 # double s5 by adding it to itself, should be 128
11 ADD $s7, $s6, $s6 # double s6 by adding it to itself, should be 256
12 J CHANGE V
14 CHANGE T: ADD $t0, $s7, $s7
15 ADD $t1, $t0, $t0
16 ADD $t2, $t1, $t1
17 ADD $t3, $t2, $t2
18 ADD $t4, $t3, $t3
19 ADD $t5, $t4, $t4
20 ADD $t6, $t5, $t5
21 ADD $t7, $t6, $t6
22 ADD $t8, $t7, $t7
23 ADD $t9, $t8, $t8
24 J CHANGE S
25
26 CHANGE A: ADD $a0, $t9, $t9
27 ADD $a1, $a0, $a0
28 ADD $a2, $a1, $a1
29 ADD $a3, $a2, $a2
30 J CHANGE S
32 CHANGE V: ADD $v0, $a3, $a3
33 ADD $v1, $v0, $v0
34 J CHANGE A
                                                                イロト イポト イヨト イヨト
```

CSci 127 (Hunter)

1 May 2018 6 / 24

WeMIPS

This is the same as the doubler, except the jumps cause the order # to change drastically, therefore all of the values will be different. CHANGE_S: ADDI \$t0, \$zero, 2 CHARGE 38 ADD to the sector 3 ADD to the sect 12 J CHANGE V 14 CHANGE_T: ADD \$t0, \$s7, \$s7 15 ADD \$t1, \$t0, \$t0 16 ADD St2, St1, St1 17 ADD \$t3, \$t2, \$t2 18 ADD \$t4, \$t3, \$t3 19 ADD \$t5, \$t4, \$t4 20 ADD \$t6, \$t5, \$t5 21 ADD \$t7, \$t6, \$t6 22 ADD \$t8, \$t7, \$t7 23 ADD \$t9, \$t8, \$t8 24 J CHANGE S 26 CHANGE A: ADD \$a0, \$t9, \$t9 27 ADD \$a1, \$a0, \$a0 28 ADD \$a2, \$a1, \$a1 29 ADD \$a3, \$a2, \$a1 30 J CHANGE S 32 CHANGE_V: ADD \$v0, \$a3, \$a3 33 ADD \$v1, \$v0, \$v0 34 J CHANGE_A

(Demo with WeMIPS)

1 May 2018 7 / 24

In Pairs or Triples:

- Write a complete Python program that converts kilograms to pounds.
- <u>Predict what the C++ code will do:</u>



イロト イポト イヨト イヨト

Python Tutor

• Write a complete **Python program** that converts kilograms to pounds.

(Write from scratch in pythonTutor.)

onlinegdb demo

1 //Another C+-program, demonstrating variables Ainclude -lostream using namespace std; 4 5 int min () 6-{ 7 int year; 6 cout << "Enter a number: "; 9 coin >> year; 10 cout << "Hello " << year << "!!\n\n"; 11 return 0; 12 }

(Demo with onlinegdb)

= nar

	<pre>//Another C++ program, demonstrating variables #include <iostream> using namespace std;</iostream></pre>
	int main () {
	int year;
	<pre>cout << "Enter a number: ";</pre>
	cin >> year;
	<pre>cout << "Hello << year << "!!\n\n";</pre>
	return 0.

CSci 127 (Hunter)

3

596

イロト イポト イヨト イヨト

• C++ is a popular programming language that extends C.

1 //Another C++ program, demonstrating variables
2 #Include clostreams
3 using nomespace std;
4 int main Q
6 int main Q
6 int space;
7 cout << "Enter a number: ";
9 cin> year;
10 cout << "Mello [! << year << "H\n\n";
11 return 0;</pre>

- C++ is a popular programming language that extends C.
- Fast, efficient, and powerful.

3

Sar



- C++ is a popular programming language that extends C.
- Fast, efficient, and powerful.
- Used for systems programming (and future courses!).



- C++ is a popular programming language that extends C.
- Fast, efficient, and powerful.
- Used for systems programming (and future courses!).
- Today, we'll introduce the basic structure and simple input/output (I/O) in C/C++.

• Programs are organized in functions.

	using namespace std;
	int main ()
	int year;
	<pre>cout << "Enter a number: ";</pre>
	cin >> year;
	<pre>cout << "Hello " << year << "!!\n\n";</pre>

996

- Programs are organized in functions.
- Variables must be declared before used:

1	
2	
3	using namespace std;
4	
5	int main ()
6 -	
7	int year;
8	cout << "Enter a number: ";
9	cin >> year;
10	<pre>cout << "Hello " << year << "!!\n\n";</pre>
11	
12	

200

- Programs are organized in functions.
- Variables must be declared before used:

int num;

1	//Another C++ program, demonstrating variables
2	
3	using namespace std;
4	
5	int main ()
6 -	
7	int year;
8	<pre>cout << "Enter a number: ";</pre>
9	cin >> year;
10	cout << "Hello " << year << "!!\n\n";
11	
12	

990

2	
3	using namespace std;
4	
5	int main ()
6- 7	
7	int year;
8	<pre>cout << "Enter a number: ";</pre>
9	cin >> year;
10	<pre>cout << "Hello << year << "!!\n\n";</pre>
11	
12	

- Programs are organized in functions.
- Variables must be declared before used:

int num;

- Many types available:
 - int, float, char, ...

200

	using namespace std;
	int main ()
	int year;
	<pre>cout << "Enter a number: ";</pre>
	cin >> year;
	<pre>cout << "Hello << year << "!!\n\n";</pre>

- Programs are organized in functions.
- Variables must be declared before used:

int num;

Many types available:

int, float, char, ...

• To print, we'll use cout <<:

Sac

	using namespace std;
	int main ()
	int year;
	<pre>cout << "Enter a number: ";</pre>
	cin >> year;
	<pre>cout << "Hello " << year << "!!\n\n";</pre>

- Programs are organized in functions.
- Variables must be declared before used:

int num;

- Many types available: int, float, char, ...
- To print, we'll use cout <<: cout << "Hello!!"</pre>

Sac

1	//Another C++ program, demonstrating variables
2	
3	using namespace std:
4	
5	int main ()
6-	
7	int year;
8	cout << "Enter a number: ":
9	cin >> vear:
10	cout << "Hello " << year << "!!\n\n";
11	
11 12	

- Programs are organized in functions.
- Variables must be **declared** before used:

int num;

- Many types available: int, float, char, ...
- To print, we'll use cout <<: cout << "Hello!!!"
- To get input, we'll use cin >>:

Sac

	//Another L++ program, demonstrating variables
	using namespace std;
	int main ()
	int year;
	<pre>cout << "Enter a number: ";</pre>
	cin >> year;
	<pre>cout << "Hello << year << "!!\n\n";</pre>

- Programs are organized in functions.
- Variables must be declared before used:

int num;

- Many types available: int, float, char, ...
- To print, we'll use cout <<: cout << "Hello!!"</pre>
- To get input, we'll use cin >>: cin >> num

Sac

	//Another C++ program, demonstrating variables
	using namespace std:
	int main ()
	int year;
	<pre>cout << "Enter a number: ";</pre>
	cin >> year;
	cout << "Hello << year << "!!\n\n";

- Programs are organized in functions.
- Variables must be **declared** before used:

int num;

- Many types available: int, float, char, ...
- To print, we'll use cout <<: cout << "Hello!!"
- To get input, we'll use cin >>: cin >> num
- To use those I/O functions, we put at the top of the program:

イロト イポト イヨト イヨト

Sac

1	<pre>//Another C++ program, demonstrating variables</pre>
2	
3	using namespace std;
4	
5	int main ()
6-	
7	int year;
8	cout << "Enter a number: ":
9	cin >> vear:
10 11 12	cout << "Hello << year << "!!\n\n";
11	
12	

- Programs are organized in functions.
- Variables must be **declared** before used:

int num;

- Many types available: int, float, char, ...
- To print, we'll use cout <<: cout << "Hello!!"
- To get input, we'll use cin >>: cin >> num
- To use those I/O functions, we put at the top of the program: #include <iostream> using namespace std;

イロト イポト イヨト イヨト

Sac

In Pairs or Triples:

Predict what the following pieces of code will do:

```
//Another C++ program, demonstrating I/O & arithmetic
#include <iostream>
using namespace std;
int main ()
{
  float kg, lbs;
  cout << "Enter kg: ";</pre>
  cin >> kg;
  1bs = kg * 2.2;
  cout << endl << "Lbs: " << lbs << "\n\n":
  return 0:
}
```

イロト 不得 トイヨト イヨト ヨー のくや

C++ Demo

```
//Another C++ program, demonstrating I/O & arithmetic
#include <iostream>
using namespace std;
```

```
int moin O
{
    flot kg, lbs;
    cout << "Enter kg: ";
    cin >> kg;
    lbs - kg 2.2;
    cost << "Lbs: " << lbs << "\n\n";
}
</pre>
```

(Demo with onlinegdb)

In Pairs or Triples:

Predict what the following pieces of code will do:

```
//Another C++ program; Demonstrates loops
#include <iostream>
using namespace std;
int main ()
ł
  int i,j;
  for (i = 0; i < 4; i++)
  {
      cout << "The world turned upside down...\n";</pre>
  }
  for (j = 10; j > 0; j - -)
  {
     cout << j << " ";
  }
  cout << "Blast off!!" << endl;</pre>
  return ∅;
}
    CSci 127 (Hunter)
                                   Lecture 12
```

= nar

15 / 24

1 May 2018

$\mathsf{C}{++} \mathsf{Demo}$

```
//Another C++ program: Demonstrates loops
#include <iostream>
using namespace std;
int main ()
  int i,j;
  for (i = 0; i < 4; i++)
  £
      cout << "The world turned upside down...\n";</pre>
  3
  for (j = 10; j > 0; j - -)
  {
      cout << j << " ";
  3
  cout << "Blast off!!" << endl;</pre>
  return 0;
3
```

(Demo with onlinegdb)

CSci 127 (Hunter)

1 May 2018 16 / 24

Sac

Definite loops

```
//Another C++ program; Demonstrates loops
#include <iostream>
using nomespace std;
int main ()
{
    int i,j;
    for (i = 0; i < 4; i++)
    {
        cout << "The world turned upside down...\n";
    }
    for (j = 10; j > 0; j--)
    {
        cout << j << "";
        cout << "Blast offl!" << endl;
        return 0;
    }
}</pre>
```

General format:

```
for ( initialization ; test ; updateAction )
{
     command1;
     command2;
     command3;
     ...
```

イロト イポト イヨト イヨト

1 May 2018 17 / 24

= nar

In Pairs or Triples:

Predict what the following pieces of code will do:

```
//Growth example
#include <iostream>
using namespace std;
int main ()
{
  int population = 100;
  cout << "Year\tPopulation\n";</pre>
  for (int year = 0; year < 100; year= year+5)
  {
      cout << year << "\t" << population << "\n";</pre>
      population = population * 2;
  }
  return ∅;
}
```

Sac

$\mathsf{C}{++} \mathsf{Demo}$

```
//forowth example
#finclud ciostreams
using namespace std;
int main ()
{
    int population = 100;
    cout << "Year\+Population\n";
    for (int year = 0; year < 100; year= year+5)
    {
        cout << year << "\t" << population = ropulation << "\n";
        population = population * 2;
    }
    return 0;
}</pre>
```

```
(Demo with C++)
```

996

In Pairs or Triples:

Predict what the following pieces of code will do:

```
//Another C++ program; Demonstrates loops
#include <iostream>
using namespace std;
int main ()
{
  int i,j,size;
  cout << "Enter size: ";</pre>
  cin >> size;
  for (i = 0; i < size; i++)
  {
    for (j = 0; j < size; j++)
      cout << "*";
    cout << endl:
  }
  cout << "\n\n";</pre>
  for (i = size; i > 0; i--)
  {
    for (j = 0; j < i; j++)
      cout << "*":
    cout << endl;
  3
  return 0;
}
     CSci 127 (Hunter)
                                        Lecture 12
```

$\mathsf{C}{++} \mathsf{Demo}$

```
//Another C++ program; Demonstrates loops
#include <iostream>
using namespace std:
int main ()
{
  int i,j,size;
  cout << "Enter size: ";</pre>
  cin >> size:
  for (i = 0; i < size; i++)</pre>
     for (j = 0; j < size; j++)
     cout << "*";
    cout << endl;
   3
  cout << "\n\n";</pre>
  for (i = size; i > 0; i--)
   {
     for (j = 0; j < i; j++)
     cout << "*":
    cout << endl:
   3
   return 0;
3
```

(Demo with C++)

Lecture Slips

In pairs or triples: **translate** the C++ program into Python:

```
//Growth example
#include <iostream>
using namespace std:
int main ()
{
  int population = 100;
  cout << "Year\tPopulation\n";</pre>
  for (int year = 0; year < 100; year= year+5)
  {
      cout << year << "\t" << population << "\n";</pre>
      population = population * 2;
  }
  return 0;
```

1 May 2018 22 / 24

E SQC

• On lecture slip, write down a topic you wish we had spent more time (and why).



590



- On lecture slip, write down a topic you wish we had spent more time (and why).
- C++ is a popular programming language that extends C.

9990



- On lecture slip, write down a topic you wish we had spent more time (and why).
- C++ is a popular programming language that extends C.
- Input/Output (I/O):
 - ▶ cin >>
 - ▶ cout <<

Sac



- On lecture slip, write down a topic you wish we had spent more time (and why).
- $\hfill \mathsf{C}++$ is a popular programming language that extends C.
- Input/Output (I/O):
 - ▶ cin >>
 - \blacktriangleright cout <<
- Definite loops:

for (i = 0; i < 10; i++)

= nar



• Lightning rounds:

990



- Lightning rounds:
 - write as much you can for 60 seconds;



- Lightning rounds:
 - write as much you can for 60 seconds;
 - ► followed by answer; and

CSci 127 (Hunter)

Lecture 12

1 May 2018 24 / 24



- Lightning rounds:
 - write as much you can for 60 seconds;
 - ► followed by answer; and
 - repeat.

CSci 127 (Hunter)

Lecture 12

1 May 2018 24 / 24

< ロ ト < 団 ト < 三 ト < 三 ト</p>



- Lightning rounds:
 - write as much you can for 60 seconds;
 - followed by answer; and
 - ► repeat.
- Continue from last time on the mock exam (on web page).

CSci 127 (Hunter)

Lecture 12

1 May 2018 24 / 24

< □ > < □ > < □ > < □ > < □ > < □ >