CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

イロト イポト イヨト イヨト

CSci 127 (Hunter)

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 1 / 36

3

990

Lecture Slips: tinyurl.com/ycrcn3k6

1	2
Of the programs 31 through 45, which did you enjoy the most?	Of the programs 31 through 45, on which did you spend the most time?
31. Shelter Population	31. Shelter Population
32. Hello, World! (main())	
33. Binary to Decimal	32. Hello, World! (main())
34. Always False Circuit	33. Binary to Decimal
35. DNA Complementary Strands	34. Always False Circuit
36. Parking Tickets	35. DNA Complementary Strands
37. Month Strings	36. Parking Tickets
38. Incrementer Circuit	37. Month Strings
39. Collision Contributing Factors	38. Incrementer Circuit
40. Github Account	39. Collision Contributing Factors
41. NYC Map	40. Github Account
42. Collisions Map	41. NYC Map
43. Copenhagen Transit Fares	42. Collisions Map
44. Closest Point	43. Copenhagen Transit Fares
44. Closest Point 5. Average Image (koalastothemax)	44. Closest Point
	45. Average Image (koalastothemax)
Other:	Other:
Why?	
Your answer	Why?
	Your answer

つへで 2 / 36

Announcements



• There's 7 additional sections for the second semester programming lab CSci 136 open for next term.

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 3 / 36

12 1

イロト イポト イヨト イ

Sac

Announcements



- There's 7 additional sections for the second semester programming lab CSci 136 open for next term.
- Today's lecturers include:
 - Prof. Sakas (department chair),
 - Genady Maryash (adjunct coordinator),
 - Katherine Howitt (tutor coordinator).

CSci 127 (Hunter)

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 3 / 36

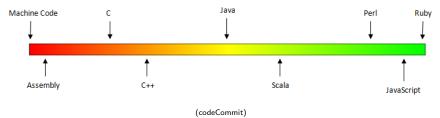
Today's Topics



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

Lecture 12: tinyurl.com/ycrcn3k6

590



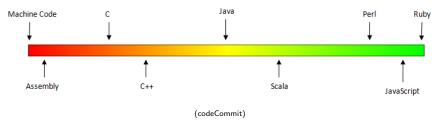
• Can view programming languages on a continuum.

CSci 127 (Hunter)

Lecture 12: tinyurl.com/ycrcn3k6

1 22 November 2017 5 / 36

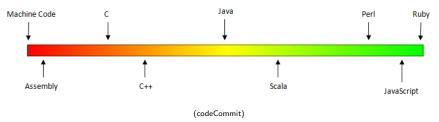
990



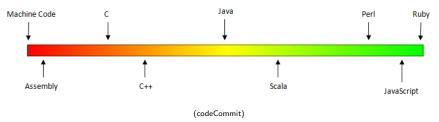
- Can view programming languages on a continuum.
- Those that directly access machine instructions & memory and have little abstraction are **low-level languages**

3

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**.

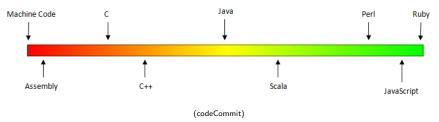
CSci 127 (Hunter)

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 5 / 36

200

イロト 不得 トイヨト イヨト 二日



- 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.

CSci 127 (Hunter)

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 5 / 36

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 MUHHABIZC 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.

CSci 127 (Hunter)

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 6 / 36

3

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.

Lecture 12: tinyurl.com/ycrcn3k6



(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.

Lecture 12: tinyurl.com/ycrcn3k6



⁽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.
- More in future architecture classes....

Lecture 12: tinyurl.com/ycrcn3k6

"Hello World!" in Simplified Machine Language

Line: 3 Go!	Show/Hide Demos							User Guide Unit T	ests Doc:
	Addition Doubler	Stav Looper	Stack Test	Hello World					
	Code Gen Save Strin	g Interactive	Binary2 Deci	imal Decir	nal2 Binary				
	Debug								
# Store 'Hello wor ADDI \$sp, \$sp, -13	ld!' at the top of	the stack				Step	Run	Enable auto switching	
ADDI \$50, \$50, -13 ADDI \$t0, \$zero, 7 SB \$t0, 0(\$50)	2 # H					s	т	A V Stack Log	
ADDI \$t0, \$zero, 1 SB \$t0, 1(\$sp)							s0:	10	
ADDI \$t0, \$zero, 1 SB \$t0, 2(\$sp)	08 # 1						su. s1:	9	
ADDI \$t0, 2(\$sp)	08 # 1								
SB \$t0, 3(\$sp)							s2:	9	
ADDI \$t0, \$zero, 1 SB \$t0, 4(\$sp)	11 # o						s3:	22	
ADDI \$t0, \$zero, 3	2 # (space)						s4:	696	
SB \$t0, 5(\$sp)							s5:	976	
ADDI \$t0, \$zero, 1 SB \$t0, 6(\$sp)	19 # W						s6:	927	
ADDI \$t0, \$zero, 1	11 # o						s7:	418	
SB \$t0, 7(\$sp)							57.	410	
ADDI \$t0, \$zero, 1 SB \$t0, 8(\$sp)	14 # r								
ADDI \$t0, \$zero, 1	08 # 1								
SB \$t0, 9(\$sp)									
ADDI \$t0, \$zero, 1 SB \$t0, 10(\$sp)	00 # d								
ADDI \$t0, Szero, 3	3 # 1								
SB \$t0, 11(\$sp)									
ADDI \$t0, \$zero, 0 SB \$t0, 12(\$sp)	# (null)								
ADDI \$v0, \$zero, 4 ADDI \$a0, \$sp, 0	# 4 is for print :	string							
2 syscall	# print to the le								

(WeMIPS)

CSci 127 (Hunter)

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 7 / 36

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
  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 St1, St0, St0
16 ADD $t2, $t1, $t1
17 ADD St3, St2, St2
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)

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 8 / 36

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)

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 9 / 36

Sac

In Pairs or Triples:

- Write a complete Python program that converts kilograms to pounds.
- Predict what the code will do:



CSci 127 (Hunter)

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 10 / 36

Sac

- - E - b-

Python Tutor

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

(Write from scratch in pythonTutor.)

CSci 127 (Hunter)

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 11 / 36

< □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □

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)

CSci 127 (Hunter)

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 12 / 36

3

Sac

2	<pre>//Another C++ program, demonstrating variables #include <iostream> using namespace std;</iostream></pre>
5	
7 8 9	<pre>int year; cout << "Enter a number: "; cin >> year:</pre>
10 11	<pre>con >> year; cout << "Hello " << year << "!!\n\n"; return 0:</pre>

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

3

900

1 //Another C++ program, demonstrating variables
2 #include-iostreams
3 using numespace std;
4 int main Q
6 (
7 (
17 var);
8 cout << "Enter a number; ";
9 cin>y var;
10 cout << "Hello ;" << ywar << "!!\n\n";
11 cout << "Hello ;" <</pre>

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

Lecture 12: tinyurl.com/ycrcn3k6



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



- C++ is a popular programming language that extends C.
- Produces 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++.

Lecture 12: tinyurl.com/ycrcn3k6

• 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>

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 14 / 36

Ξ

996

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

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	<pre>cout << "Hello " << year << "!!\n\n";</pre>
11	
12	

3

596

- 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	

3

596

1	
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, ...

Lecture 12: tinyurl.com/ycrcn3k6

= nar

	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 <<:

3

SOC

	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!!"

= nar

	<pre>//Another C++ program, demonstrating variables</pre>
	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 >>:

Lecture 12: tinyurl.com/ycrcn3k6

Sac

	//Another C++ program, demonstrating variables
	using namespace std;
	int main ()
	int year;
	<pre>cout << "Enter a number: ";</pre>
	cin >> year;
I	<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!!"
- To get input, we'll use cin >>: cin >> num

Lecture 12: tinyurl.com/ycrcn3k6

Sac

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	<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, ...
- 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

Lecture 12: tinyurl.com/ycrcn3k6

Sac

	//Another C++ 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!!"
- 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

Lecture 12: tinyurl.com/ycrcn3k6

Sac

In Pairs or Triples:

Predict what the following pieces of code will do:

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

```
return 0;
```

```
•
```

```
CSci 127 (Hunter)
```

cout << "Blast off!!" << endl;</pre>

Lecture 12: tinyurl.com/ycrcn3k6

= nar

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

```
//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;
    cos - dedd << "Lbs: " << lbs << "\n\n";
}</pre>
```

(Demo with onlinegdb)

CSci 127 (Hunter)

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 16 / 36

イロト イポト イヨト イヨト ヨー つくや

In Pairs or Triples:

Predict what the following pieces of code will do:

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

```
//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":
  for (i = size: i > 0: i--)
  {
    for (j = 0; j < i; j++)
    cout << "*";
    cout << endl;
  return 0:
```

イロト イポト イヨト イヨト

CSci 127 (Hunter)

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 17 / 36

= nar

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

```
//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++)
```

CSci 127 (Hunter)

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 18 / 36

イロト イロト イヨト イヨト

Definite loops

```
//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++)</pre>
    cout << "*":
    cout << endl:
  cout << "\n\n":</pre>
  for (i = size; i > 0; i--)
    for (j = 0; j < i; j++)
    cout << "*":
    cout << endl:
  return 0:
```

```
General format:
```

for (initialization ; test ; updateAction)

command1; command2; command3;

. . .

CSci 127 (Hunter)

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 19 / 36

3

Sac

イロト イポト イヨト イヨト

Final Overview (Cont'd from Last Lecture)

For each question below, first write **the function header (name & inputs) and return values** (often called the Application Programming Interface (API)). Then write the complete function:

- Write a function that takes a whole number and returns the corresponding binary number as a string.
- Write a function that takes a weight in kilograms and returns the weight in pounds.
- Write a function that, given a DataFrame, returns the minimal value in the "Manhattan" column.
- Write a function that computes the total monthly payment when given the initial loan amount, annual interest rate, number of years of the loan.
- Write a function that takes a string and returns its length.

(Hint: highlight key words, make list of inputs, list of outputs, then put together.)

CSci 127 (Hunter)

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 20 / 36

For each question below, write the function header (name & inputs) and return values (often called the Application Programming Interface (API)):

• Write a function that takes a whole number and returns the corresponding binary number as a string.

def num2bin(num):

```
...
return(bin)
```

イロト 不得 トイヨト イヨト ヨー のくや

For each question below, write the function header (name & inputs) and return values (often called the Application Programming Interface (API)):

• Write a function that takes a whole number and returns the corresponding binary number as a string.

```
def num2bin(num):
    binStr = ""
    while (num > 0):
        #Divide by 2, and add the remainder to the string
        r = num %2
        binString = str(r) + binStr
        num = num / 2
    return(binStr)
```

CSci 127 (Hunter)

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 22 / 36

For each question below, write the function header (name & inputs) and return values (often called the Application Programming Interface (API)):

• Write a function that takes a weight in kilograms and returns the weight in pounds.

For each question below, write the function header (name & inputs) and return values (often called the Application Programming Interface (API)):

 Write a function that takes a weight in kilograms and returns the weight in pounds.

def kg2lbs(kg):

```
...
return(lbs)
```

For each question below, write the function header (name & inputs) and return values (often called the Application Programming Interface (API)):

 Write a function that takes a weight in kilograms and returns the weight in pounds.

def kg2lbs(kg)
 lbs = kg * 2.2
 return(lbs)

Lecture 12: tinyurl.com/ycrcn3k6

For each question below, write the function header (name & inputs) and return values (often called the Application Programming Interface (API)):

• Write a function that, given a DataFrame, returns the minimal value in the "Manhattan" column.

For each question below, write the function header (name & inputs) and return values (often called the Application Programming Interface (API)):

• Write a function that, given a DataFrame, returns the minimal value in the "Manhattan" column.

def getMin(df):
 ...

return(min)

イロト 不得 トイヨト イヨト ヨー のくや

For each question below, write the function header (name & inputs) and return values (often called the Application Programming Interface (API)):

• Write a function that, given a DataFrame, returns the minimal value in the "Manhattan" column.

```
def getMin(df):
    mM = df['Manhattan'].min()
    return(mM)
```

For each question below, write the function header (name & inputs) and return values (often called the Application Programming Interface (API)):

• Write a function that computes the total monthly payment when given the initial loan amount, annual interest rate, number of years of the loan.

Sac

イロト イポト イヨト イヨト 二日

For each question below, write the function header (name & inputs) and return values (often called the Application Programming Interface (API)):

• Write a function that computes the total monthly payment when given the initial loan amount, annual interest rate, number of years of the loan.

def computePayment(loan,rate,year):

```
....
return(payment)
```

イロト 不得下 イヨト イヨト 二日

For each question below, write the function header (name & inputs) and return values (often called the Application Programming Interface (API)):

• Write a function that computes the total monthly payment when given the initial loan amount, annual interest rate, number of years of the loan.

def computePayment(loan,rate,year):
 (Some formula for payment)
 return(payment)

イロト 不得 トイヨト イヨト ヨー シタウ

For each question below, write the function header (name & inputs) and return values (often called the Application Programming Interface (API)):

• Write a function that takes a string and returns its length.

```
def sLength(str):
```

```
...
return(length)
```

▲□▶ ▲□▶ ▲□▶ ▲□▶ □ − ∽ Q (~

For each question below, write the function header (name & inputs) and return values (often called the Application Programming Interface (API)):

• Write a function that takes a string and returns its length.

```
def sLength(str):
    length = len(str)
    return(length)
```

Recap: C++

 $\label{eq:constraints} $$ Constraints loops $$ include closes exactly include closes exac$

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

CSci 127 (Hunter)

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 34 / 36

∃ <\0<</p>

イロト イロト イヨト イヨト

Recap: C++

```
//Another (-+ program, Demonstrates loops

biclude -(ostrates)

unity monspace atd;

in teal O

\begin{cases} mt tois O

model of the observation of the observation of the observation

for (1 = 0 i < 4; i++)

i cout << "The world turned upside down...\n";

for (j = 10; j > 0; j--)

i cout << "Illost off!!" << endl;

returm 0;

} returm 0;
```

- C++ is a popular programming language that extends C.
- Input/Output (I/O):
 - ▶ cin >>
 - ▶ cout <<</p>

CSci 127 (Hunter)

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 34 / 36

= nar

イロト イポト イヨト イヨト

Recap: C++

- C++ is a popular programming language that extends C.
- Input/Output (I/O):
 - ▶ cin >>
 - ▶ cout <<</p>
- Definite loops: for (i = 0; i < 10; i++)</pre>

Lecture 12: tinyurl.com/ycrcn3k6

22 November 2017 34 / 36

= nar

イロト イポト イヨト イヨト

Lecture Slips: tinyurl.com/ycrcn3k6

1	2
Of the programs 31 through 45, which did you enjoy the most?	Of the programs 31 through 45, on which did you spend the most time?
31. Shelter Population	most time?
32. Hello, World! (main())	
33. Binary to Decimal	32. Hello, World! (main())
34. Always False Circuit	33. Binary to Decimal
35. DNA Complementary Strands	34. Always False Circuit
36. Parking Tickets	35. DNA Complementary Strands
37. Month Strings	36. Parking Tickets
38. Incrementer Circuit	37. Month Strings
39. Collision Contributing Factors	38. Incrementer Circuit
40. Github Account	39. Collision Contributing Factors
41. NYC Map	40. Github Account
	41. NYC Map
42. Collisions Map	42. Collisions Map
43. Copenhagen Transit Fares	43. Copenhagen Transit Fares
44. Closest Point	44. Closest Point
45. Average Image (koalastothemax)	45. Average Image (koalastothemax)
Other:	
Why?	ourer.
Your answer	Why?
	Your answer

୬ < ୯ 35 / 36