CSCI 200: Foundational Programming Concepts & Design Lecture 00

Welcome to CSCI 200

Learning Outcomes For Today

- Recite the expectations for this course
- Discuss the differences between programming languages, specifically Python & C++
- Build your programming environment

On Tap For Today

- Syllabus
- Programming Languages
- Python vs. C++
- Practice

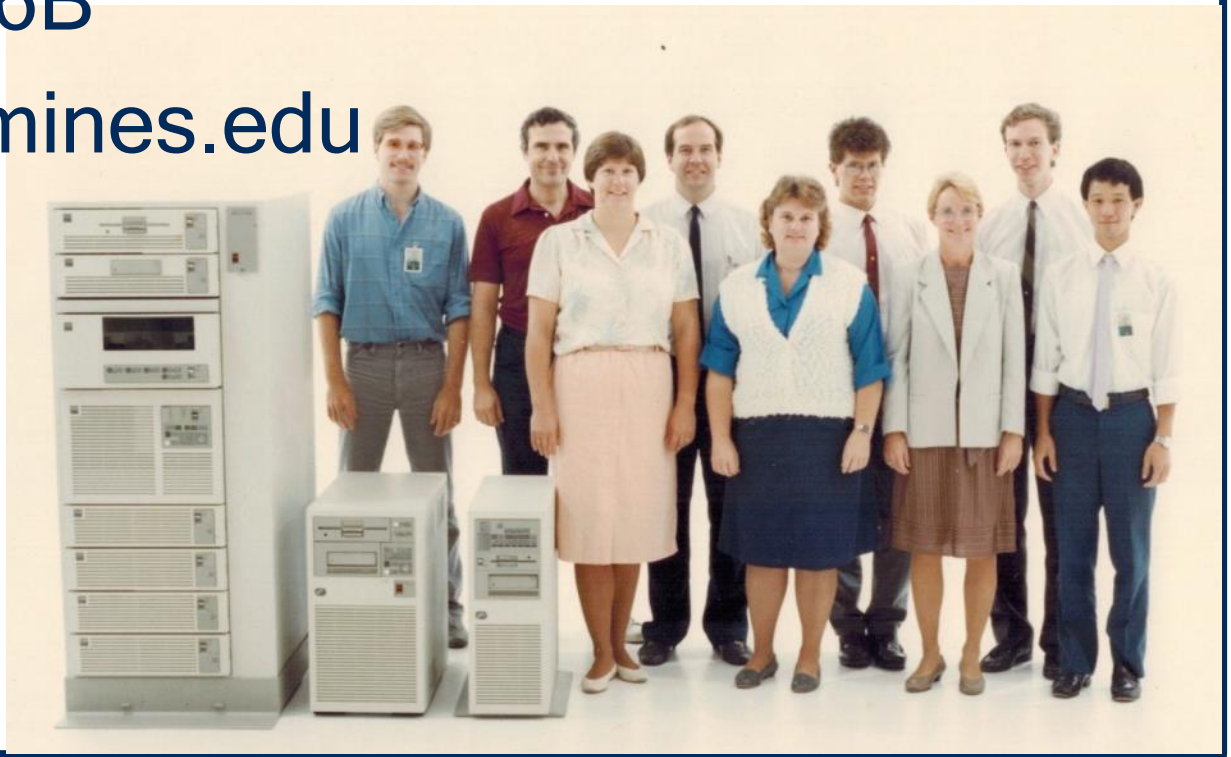
On Tap For Today

- Syllabus
- Programming Languages
- Python vs. C++
- Practice

Clark Scholten

- M/W 10:00a – 11:50a F 1p – 3p
  or by appointment

  CTLM 246B
- cscholten@mines.edu



Tell Me About Yourself!

- Who are you?
- What do you like?

- http://bit.ly/200Spring2024

What's Online

- Course Website
  - https://cs-courses.mines.edu/csci200/
    - Bookmark this page!
    - Descriptions for all labs & assignments
    - Links to all other sites
- Canvas
  - Submissions, quizzes, lecture files, etc
  - Integrates all other sites
- zyBooks
  - Interactive course eBook
- Ed Discussion
  - Out of class discussion & main source of communication

Syllabus

- Read the syllabus on course website under Files > Documents

https://cs-courses.mines.edu/csci200/resources.html

zyBook Logistics

- Do the assigned sections BEFORE coming to class
  - Some Activities required
  - Optional sections are optional
- Purpose of activities: learning not testing
  - NO penalty for an initial wrong answer
  - Unlimited attempts at each activity
- Worth 5% of your final grade
  - (should be "free" points)

zyBook Grade

- If you complete __ then your grade is __
  - >95% 100%
  - >90% 95%
  - >85% 90%
  - … …
  - >10% 15%
  - > 5% 10%
  - > 0% 5%
  - 0% 0%

How To Succeed In This Class

- Come to class
- Complete zyBook activities before class (due with HW Set)
- Complete worksheets and participation review exercises after class
- 2-3 Labs per set (due with HW Set) (14 total - averages 1 a week)
- Come to class
- 1 HW assignment every 2-3 weeks (due with HW Set) (6 total)
- 6 quizzes (in class 01/29, 02/07, 03/01, 03/08, 04/15, 04/26)
- Come to class
- 3 written exams (2 midterms in class 02/09 & 03/15 and 1 final 05/06)
- 1 semester project (out 02/26 - due 05/02)

HW Set Logistics

- Submit to canvas:
  - Labs: zip all files for each individually
  - Assignments: zip all files for each individually
- Rubric:
  - Submitted on time (accepted for 72 hrs beyond due)
  - Builds and runs without errors or warnings
  - Generates correct results
  - Style guidelines and best practices followed

Assignment Logistics

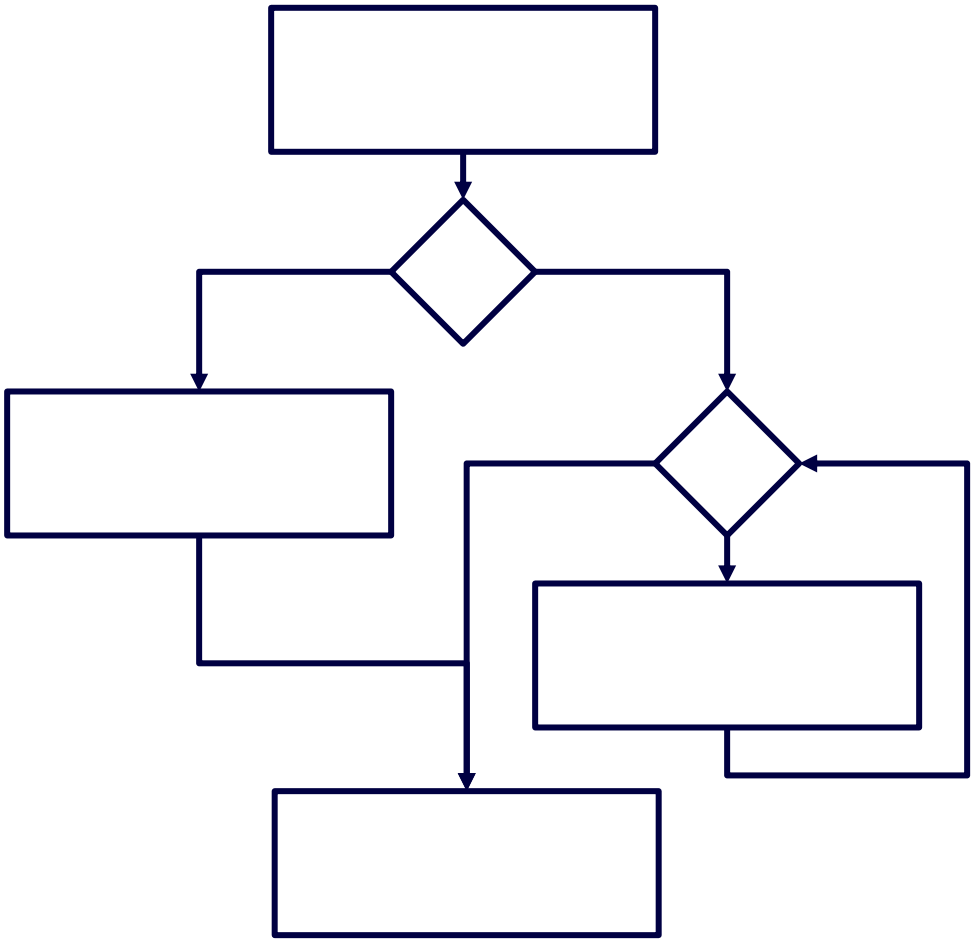
- Submissions MUST compile in the lab environment (Windows using g++)
- Submissions MUST include all necessary source code files (*.h *.cpp Makefile)

- Resubmission Policy
  - Once grades returned, may resubmit a corrected/updated version to earn up to half deducted points back

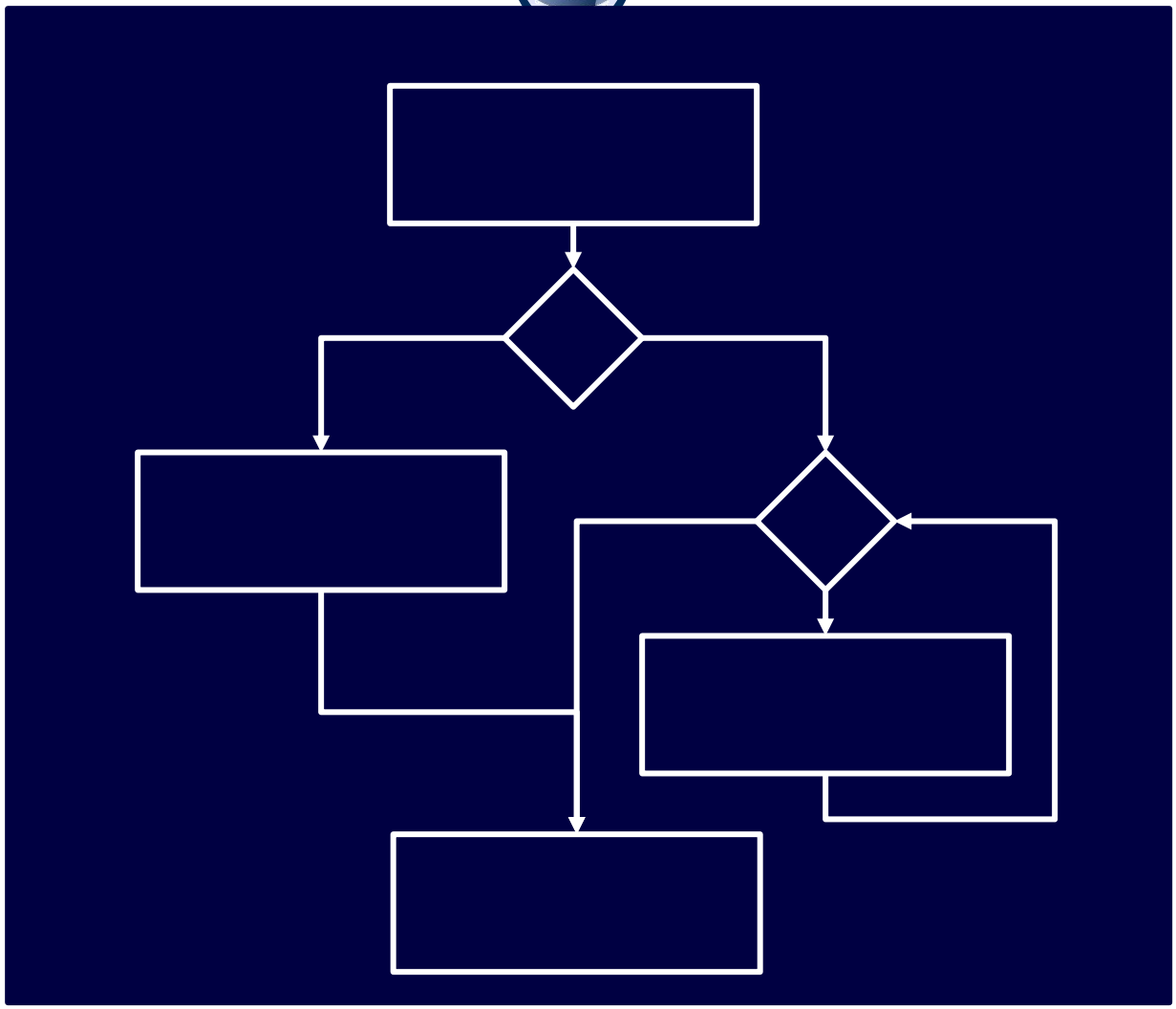
Learning Outcomes for the Course

- Design an algorithm to solve a problem by breaking the overarching problem into smaller modular components using abstraction and object-oriented design with inheritance.
- Translate the algorithm into a program using proper C++ syntax and fundamental programming constructs (e.g. control structures, I/O, classes).
- Recite & apply frequently used Linux command line commands and compile a program using a command line build system.
- Diagram memory usage, dynamically allocate & deallocate objects at run-time using "The Big Three," and trace the call stack of a program's run-time.
- Define recursion and construct common recursive data structures (e.g. linked list, stack, queue) & algorithms (e.g. search & sort).
- Diagram & construct dynamically allocated data structures (e.g. array, vector, string), recursive data structures, (e.g. linked list, stack, queue), and implement common list operations (e.g. traversal, insertion, removal)
- Define "Big-O" notation, list complexities in increasing order, and analyze an algorithm to compute its run-time performance & memory complexities

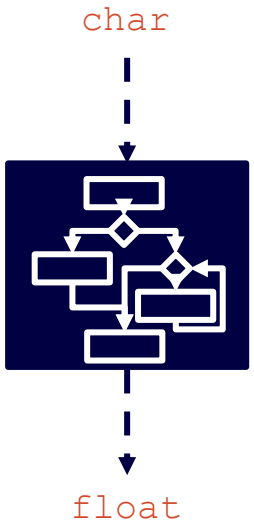
Big Picture – Program Flow



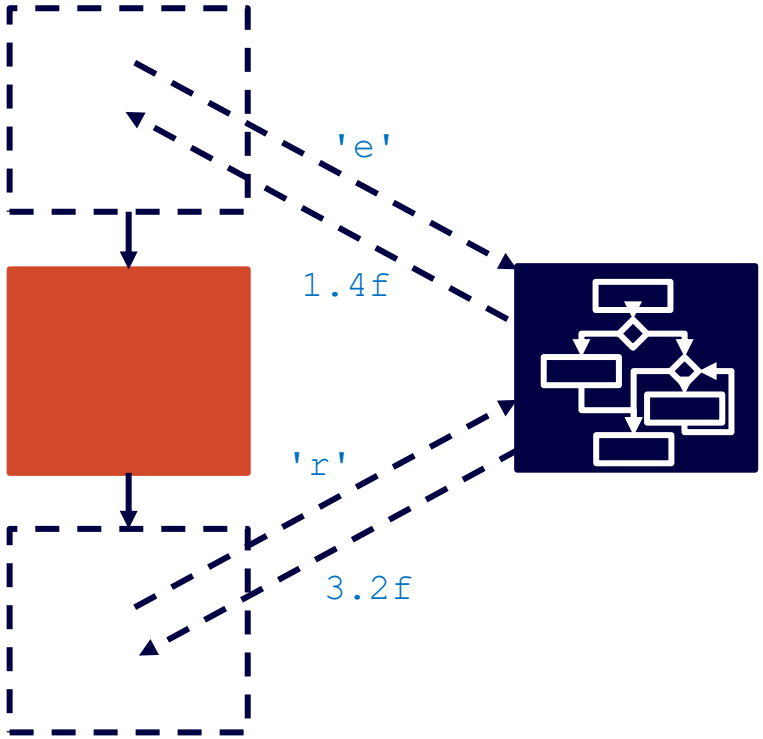
Big Picture – Program Flow



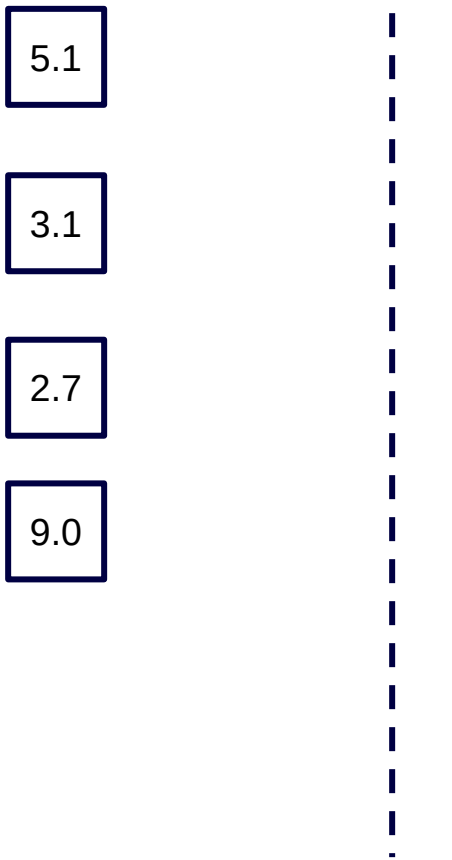
Big Picture – Program Flow



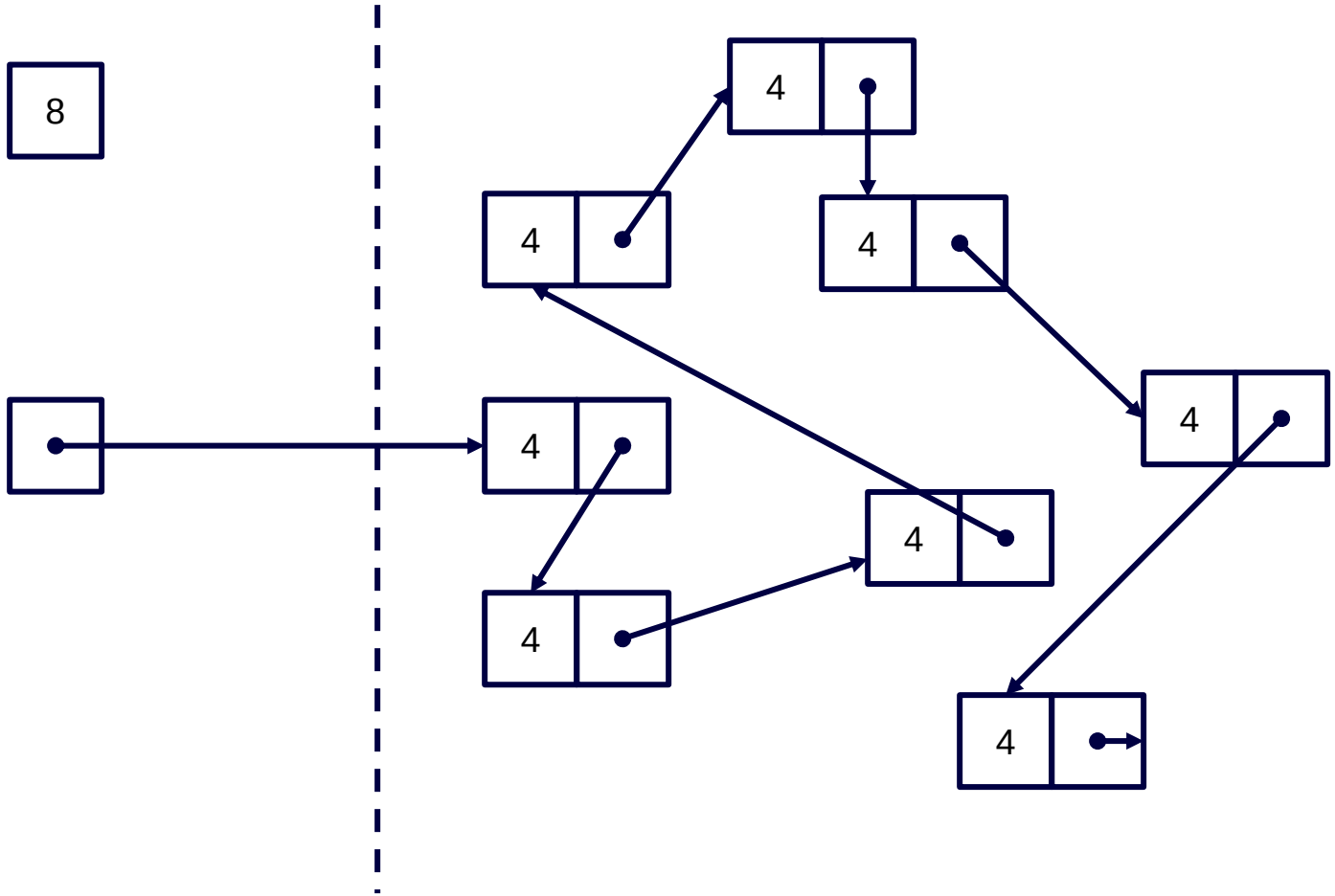
Big Picture – Program Flow



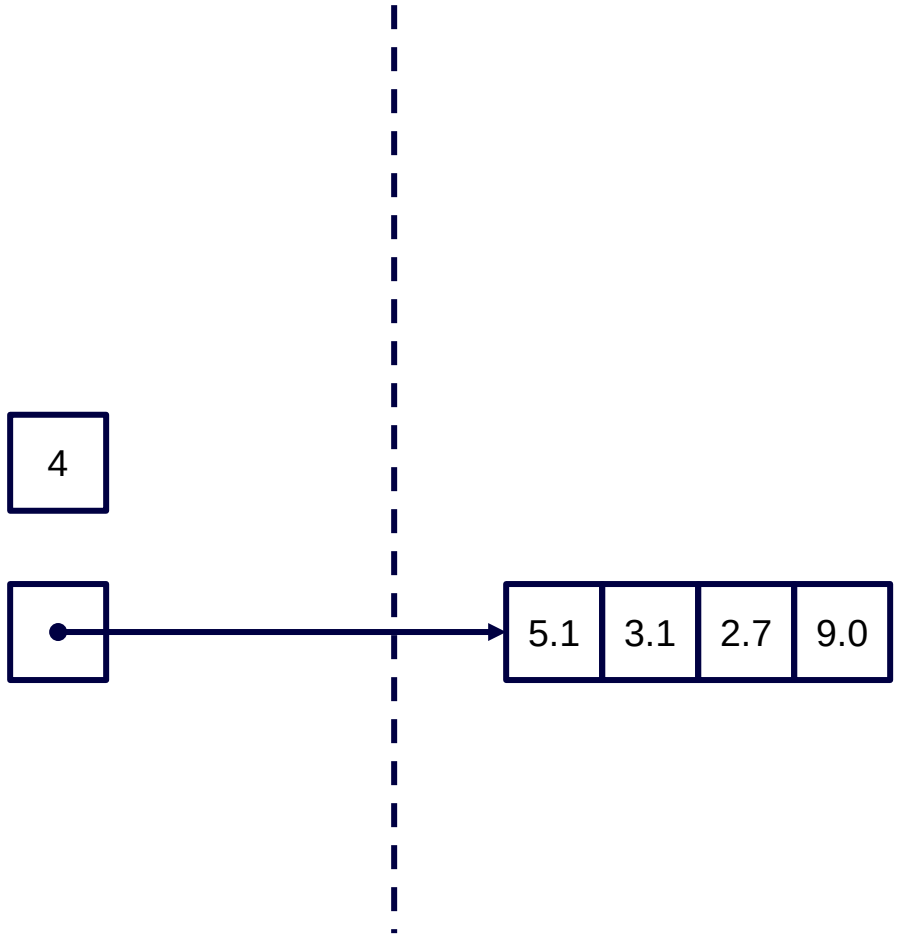
Big Picture – Data Storage



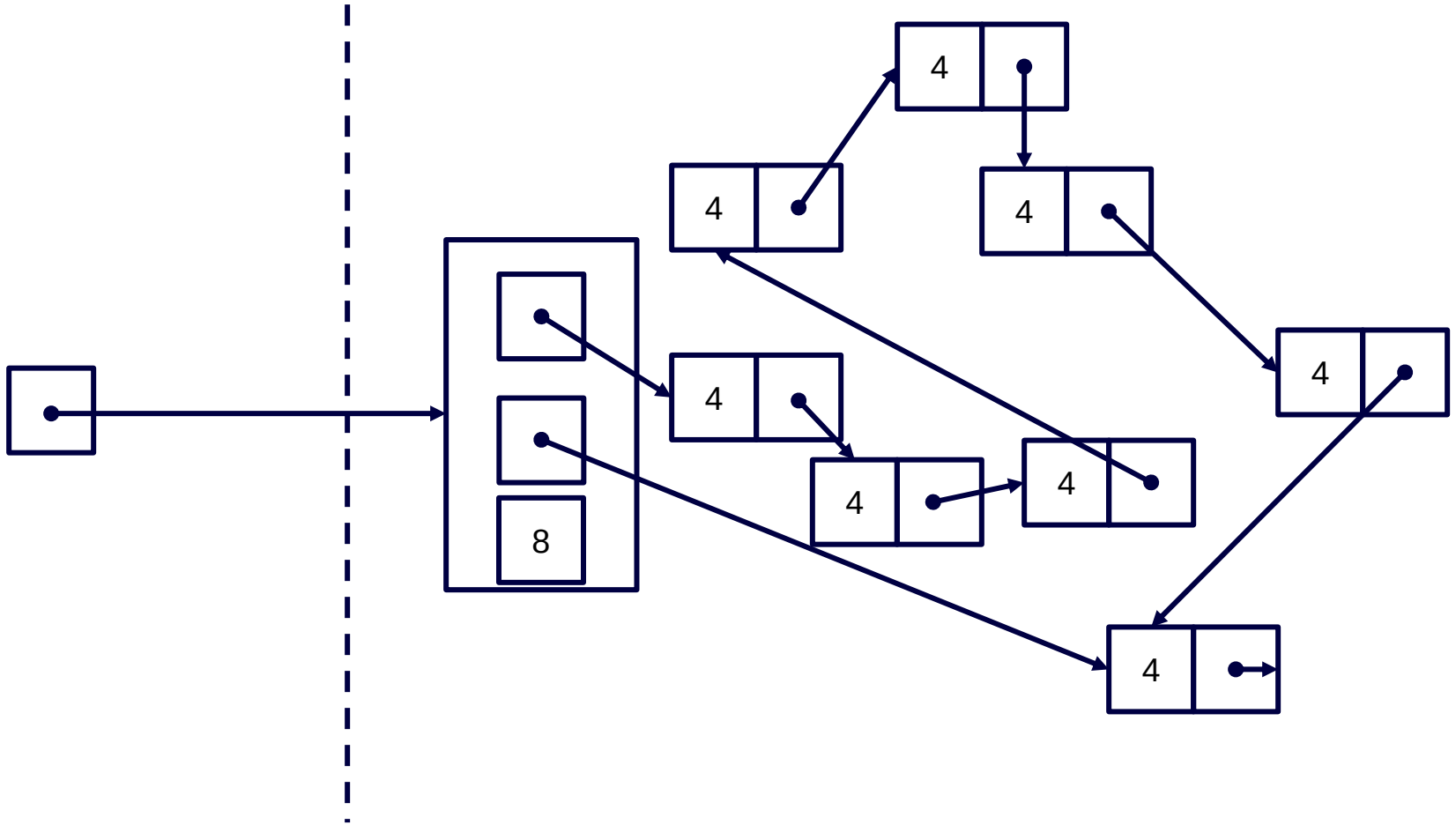
Big Picture – Data Storage



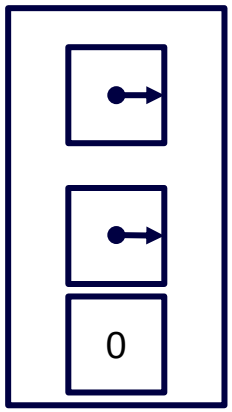
Big Picture – Data Storage



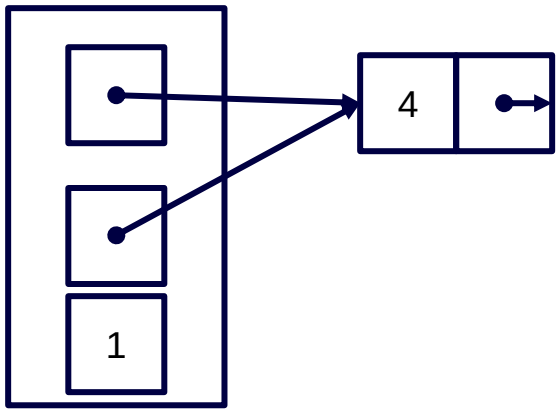
Big Picture – Data Storage



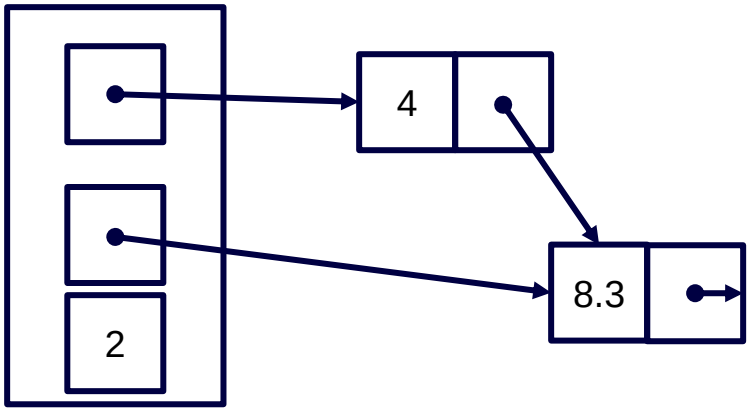
Big Picture – Class Structure



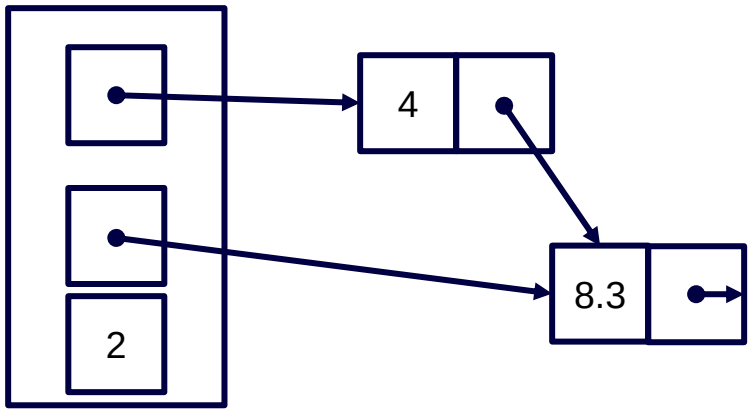
Big Picture – Class Structure



Big Picture – Class Structure

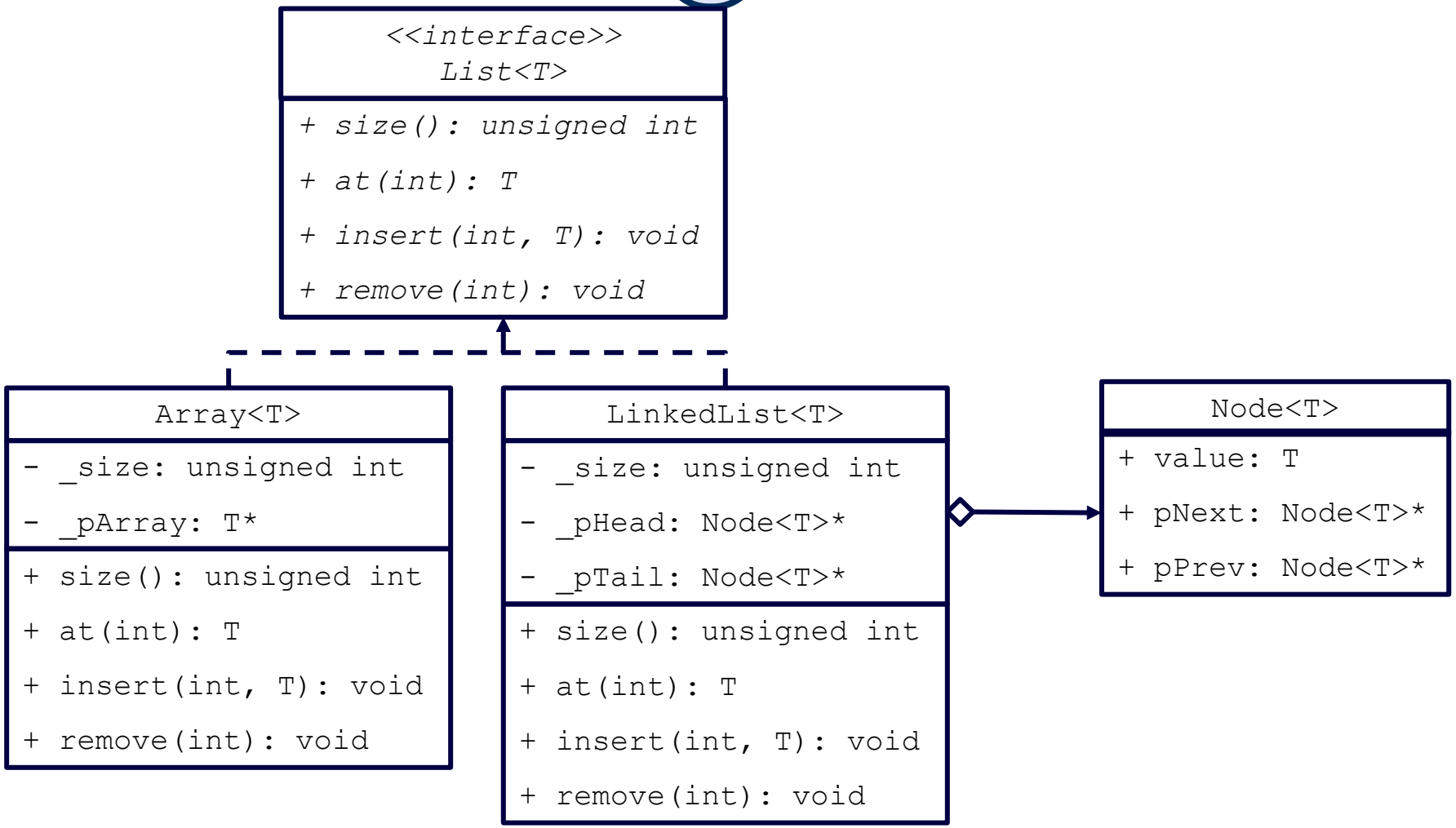


Big Picture – Class Structure



LinkedList<double>
- _size: unsigned int - _pHead: Node<double>* - _pTail: Node<double>*
+ size(): unsigned int + at(int): double + insert(int, double): void + remove(int): void

Big Picture – Class Structure



On Tap For Today

- Syllabus
- Programming Languages
- Python vs. C++
- Practice

A Brief History

- In 2024, this is a computer

- In 1623, this was a computer

René Descartes (1596 – 1650)

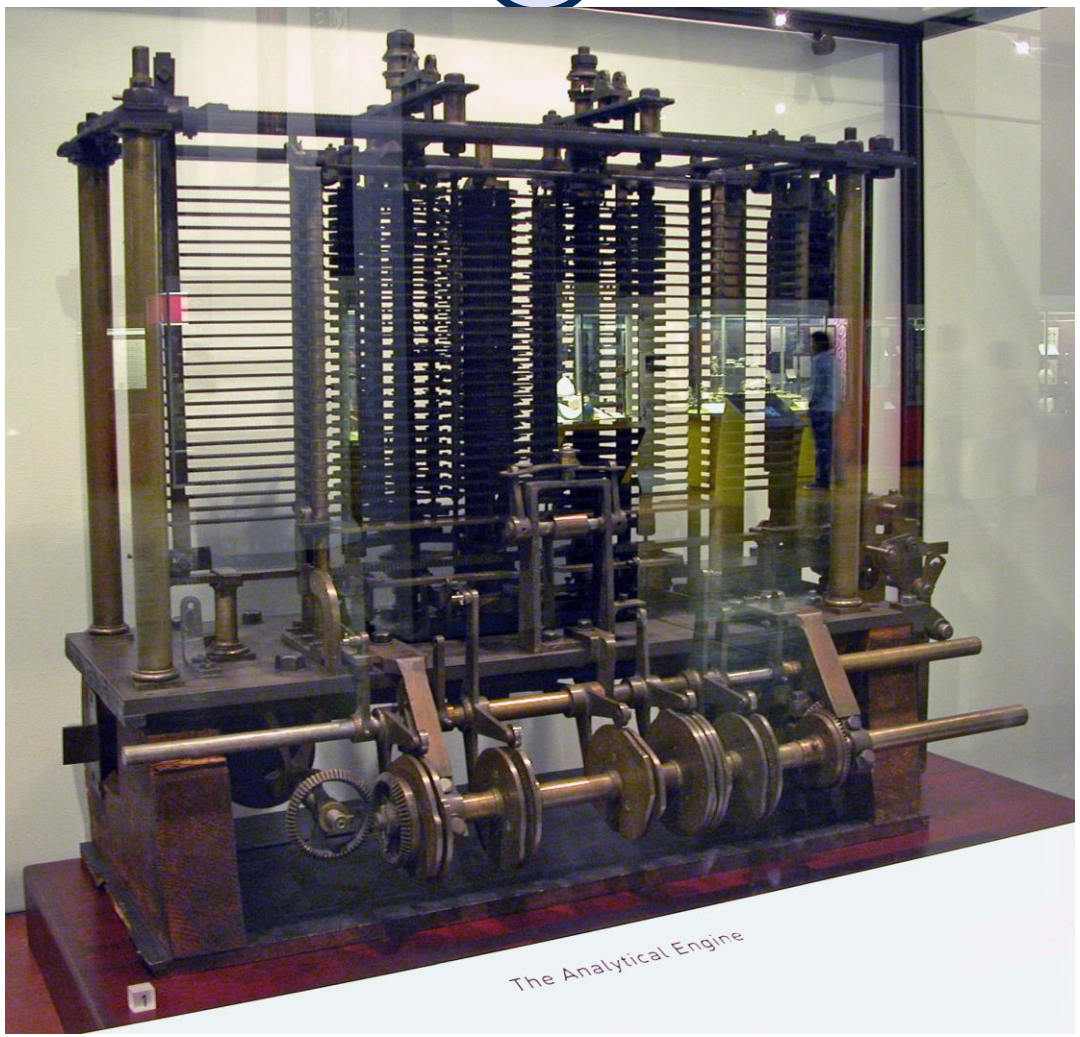
Mathematician – pioneered Cartesian geometry and standard notation of exponents

aka computer – a person that computes

Another computer

- Ada Lovelace
  - 1815-1852
- The FIRST computer programmer
- Developed an algorithm to compute Bernoulli numbers on Babbage's Analytical Machine

Babbage's Analytical Machine



Lovelace's Algorithm

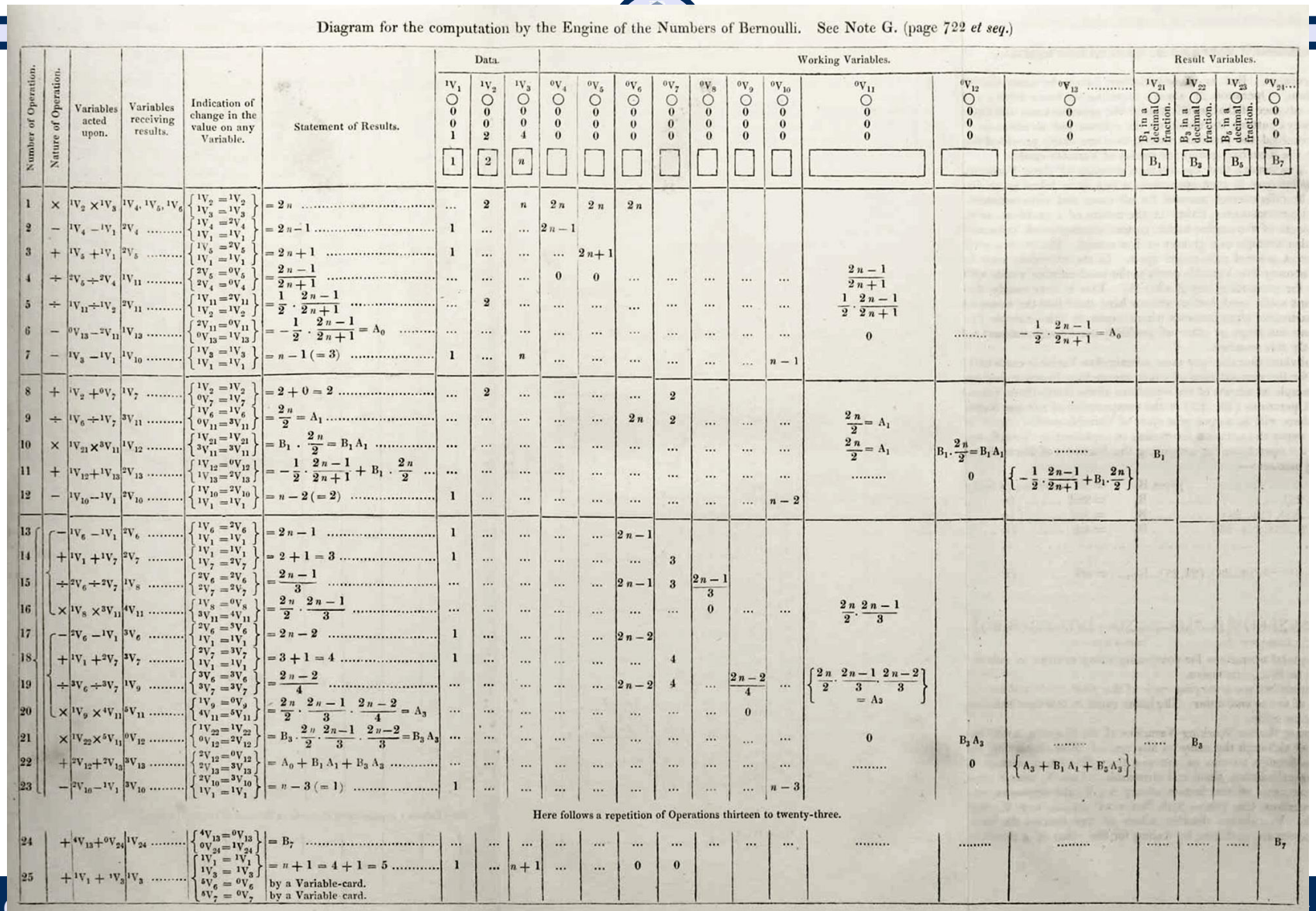


Diagram for the computation by the Engine of the Numbers of Bernoulli. See Note G. (page 722 *et seq.*)

Number of Operation.	Nature of Operation.	Variables acted upon.	Variables receiving results.	Indication of change in the value on any Variable.	Statement of Results.	Data. ${}^1V_1$ 0 0 0 1 [1]	Data. ${}^1V_2$ 0 0 0 2 [2]	Data. ${}^1V_3$ 0 0 0 4 [n]	Working Variables. ${}^0V_4$ 0 0 0 0 [ ]	${}^0V_5$ 0 0 0 0 [ ]	${}^0V_6$ 0 0 0 0 [ ]	${}^0V_7$ 0 0 0 0 [ ]	${}^0V_8$ 0 0 0 0 [ ]	${}^0V_9$ 0 0 0 0 [ ]	${}^0V_{10}$ 0 0 0 0 [ ]	${}^0V_{11}$ 0 0 0 0 [ ]	${}^0V_{12}$ 0 0 0 0 [ ]	${}^0V_{13}$ ..... 0 0 0 0 [ ]	Result Variables. ${}^1V_{21}$ B_1 in a decimal fraction. [B_1]	${}^1V_{22}$ B_3 in a decimal fraction. [B_3]	${}^1V_{23}$ B_5 in a decimal fraction. [B_5]	${}^0V_{24}$... 0 0 0 [B_7]
1	×	${}^1V_2 \times {}^1V_3$	${}^1V_4, {}^1V_5, {}^1V_6$	${}^1V_2 = {}^1V_2$, ${}^1V_3 = {}^1V_3$	$= 2n$	...	2	n	$2n$	$2n$	$2n$											
2	−	${}^1V_4 - {}^1V_1$	${}^2V_4$	${}^1V_4 = {}^2V_4$, ${}^1V_1 = {}^1V_1$	$= 2n-1$	1	...	...	$2n-1$													
3	+	${}^1V_5 + {}^1V_1$	${}^2V_5$	${}^1V_5 = {}^2V_5$, ${}^1V_1 = {}^1V_1$	$= 2n+1$	1	...	...	...	$2n+1$												
4	÷	${}^2V_5 \div {}^2V_4$	${}^1V_{11}$	${}^2V_5 = {}^0V_5$, ${}^2V_4 = {}^0V_4$	$= \frac{2n-1}{2n+1}$	...	...	...	0	0	...	...	...	...	...	$\frac{2n-1}{2n+1}$						
5	÷	${}^1V_{11} \div {}^1V_2$	${}^2V_{11}$	${}^1V_{11} = {}^2V_{11}$, ${}^1V_2 = {}^1V_2$	$= \frac{1}{2} \cdot \frac{2n-1}{2n+1}$	...	2	...	...	...	...	...	...	...	...	$\frac{1}{2} \cdot \frac{2n-1}{2n+1}$						
6	−	${}^0V_{13} - {}^2V_{11}$	${}^1V_{13}$	${}^2V_{11} = {}^0V_{11}$, ${}^0V_{13} = {}^1V_{13}$	$= -\frac{1}{2} \cdot \frac{2n-1}{2n+1} = A_0$	...	...	...	...	...	...	...	...	...	...	0	.....	$-\frac{1}{2} \cdot \frac{2n-1}{2n+1} = A_0$				
7	−	${}^1V_3 - {}^1V_1$	${}^1V_{10}$	${}^1V_3 = {}^1V_3$, ${}^1V_1 = {}^1V_1$	$= n-1\ (=3)$	1	...	n	...	...	...	...	...	...	$n-1$							
8	+	${}^1V_2 + {}^0V_7$	${}^1V_7$	${}^1V_2 = {}^1V_2$, ${}^0V_7 = {}^1V_7$	$= 2+0=2$	...	2	...	...	...	...	2										
9	÷	${}^1V_6 \div {}^1V_7$	${}^3V_{11}$	${}^1V_6 = {}^1V_6$, ${}^0V_{11} = {}^3V_{11}$	$= \frac{2n}{2} = A_1$	...	...	...	...	...	$2n$	2	...	...	...	$\frac{2n}{2} = A_1$						
10	×	${}^1V_{21} \times {}^3V_{11}$	${}^1V_{12}$	${}^1V_{21} = {}^1V_{21}$, ${}^3V_{11} = {}^3V_{11}$	$= B_1 \cdot \frac{2n}{2} = B_1 A_1$	...	...	...	...	...	...	...	...	...	...	$\frac{2n}{2} = A_1$	$B_1 \cdot \frac{2n}{2} = B_1 A_1$	.....	B_1			
11	+	${}^1V_{12} + {}^1V_{13}$	${}^2V_{13}$	${}^1V_{12} = {}^0V_{12}$, ${}^1V_{13} = {}^2V_{13}$	$= -\frac{1}{2} \cdot \frac{2n-1}{2n+1} + B_1 \cdot \frac{2n}{2}$ ...	...	...	...	...	...	...	...	...	...	...	.....	0	$\left\{-\frac{1}{2} \cdot \frac{2n-1}{2n+1} + B_1 \cdot \frac{2n}{2}\right\}$				
12	−	${}^1V_{10} - {}^1V_1$	${}^2V_{10}$	${}^1V_{10} = {}^2V_{10}$, ${}^1V_1 = {}^1V_1$	$= n-2\ (=2)$	1	...	...	...	...	...	...	...	...	$n-2$							
13	−	${}^1V_6 - {}^1V_1$	${}^2V_6$	${}^1V_6 = {}^2V_6$, ${}^1V_1 = {}^1V_1$	$= 2n-1$	1	...	...	...	...	$2n-1$											
14	+	${}^1V_1 + {}^1V_7$	${}^2V_7$	${}^1V_1 = {}^1V_1$, ${}^1V_7 = {}^2V_7$	$= 2+1=3$	1	...	...	...	...	...	3										
15	÷	${}^2V_6 \div {}^2V_7$	${}^1V_8$	${}^2V_6 = {}^2V_6$, ${}^2V_7 = {}^2V_7$	$= \frac{2n-1}{3}$	...	...	...	...	...	$2n-1$	3	$\frac{2n-1}{3}$									
16	×	${}^1V_8 \times {}^3V_{11}$	${}^4V_{11}$	${}^1V_8 = {}^0V_8$, ${}^3V_{11} = {}^4V_{11}$	$= \frac{2n}{2} \cdot \frac{2n-1}{3}$	...	...	...	...	...	...	...	0	...	...	$\frac{2n}{2} \cdot \frac{2n-1}{3}$						
17	−	${}^2V_6 - {}^1V_1$	${}^3V_6$	${}^2V_6 = {}^3V_6$, ${}^1V_1 = {}^1V_1$	$= 2n-2$	1	...	...	...	...	$2n-2$											
18	+	${}^1V_1 + {}^2V_7$	${}^3V_7$	${}^2V_7 = {}^3V_7$, ${}^1V_1 = {}^1V_1$	$= 3+1=4$	1	...	...	...	...	...	4										
19	÷	${}^3V_6 \div {}^3V_7$	${}^1V_9$	${}^3V_6 = {}^3V_6$, ${}^3V_7 = {}^3V_7$	$= \frac{2n-2}{4}$	...	...	...	...	...	$2n-2$	4	...	$\frac{2n-2}{4}$	...	$\left\{\frac{2n}{2} \cdot \frac{2n-1}{3} \cdot \frac{2n-2}{3} = A_3\right\}$							
20	×	${}^1V_9 \times {}^4V_{11}$	${}^5V_{11}$	${}^1V_9 = {}^0V_9$, ${}^4V_{11} = {}^5V_{11}$	$= \frac{2n}{2} \cdot \frac{2n-1}{3} \cdot \frac{2n-2}{4} = A_3$ ...	...	...	...	...	...	...	...	...	...	0							
21	×	${}^1V_{22} \times {}^5V_{11}$	${}^0V_{12}$	${}^1V_{22} = {}^1V_{22}$, ${}^0V_{12} = {}^2V_{12}$	$= B_3 \cdot \frac{2n}{2} \cdot \frac{2n-1}{3} \cdot \frac{2n-2}{3} = B_3 A_3$	...	...	...	...	...	...	...	...	...	...	0	$B_3 A_3$	.....	.....	B_3		
22	+	${}^2V_{12} + {}^2V_{13}$	${}^3V_{13}$	${}^2V_{12} = {}^0V_{12}$, ${}^2V_{13} = {}^3V_{13}$	$= A_0 + B_1 A_1 + B_3 A_3$	...	...	...	...	...	...	...	...	...	...	.....	0	$\{A_3 + B_1 A_1 + B_3 A_3\}$				
23	−	${}^2V_{10} - {}^1V_1$	${}^3V_{10}$	${}^2V_{10} = {}^3V_{10}$, ${}^1V_1 = {}^1V_1$	$= n-3\ (=1)$	1	...	...	...	...	...	...	...	...	$n-3$							

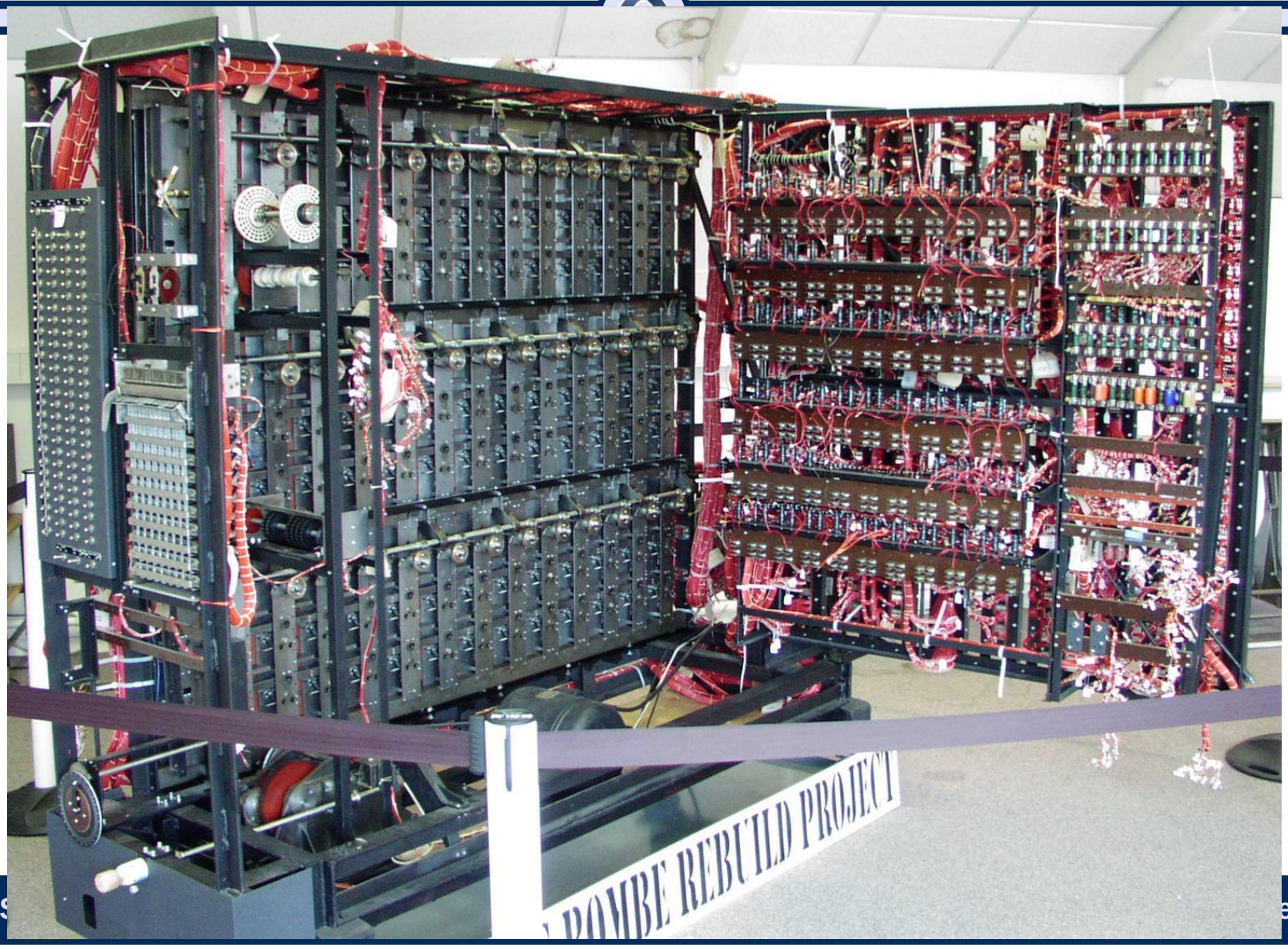
Here follows a repetition of Operations thirteen to twenty-three.

24	+	${}^4V_{13} + {}^0V_{24}$	${}^1V_{24}$	${}^4V_{13} = {}^0V_{13}$, ${}^0V_{24} = {}^1V_{24}$	$= B_7$	...	...	...	...	...	...	...	...	...	...	.....	.....	.....	.....	.....	.....	B_7
25	+	${}^1V_1 + {}^1V_3$	${}^1V_3$	${}^1V_1 = {}^1V_1$, ${}^1V_3 = {}^1V_3$, ${}^5V_6 = {}^0V_6$, ${}^5V_7 = {}^0V_7$	$= n+1 = 4+1=5$ ..... by a Variable-card. by a Variable card.	1	...	$n+1$	...	...	0	0										

More recently

- Alan Turing
  - 1912 - 1954
- Father of modern computing
  - And star of

Turing's Bombe Machine



Software Languages Since Then

- Machine Language
  - Binary Instructions: 01101101 00110110
- Assembly Language
  - Specific to the CPU
  - Manipulates the internal component
- High-Level Languages
  - C, C++, Ada, Fortran, Python, Java, Kotlin, Swift, Ruby, Rust, and many others

x86 Assembler

```
:
:
fldl   -0x28(%ebp)
fmull  -0x20(%ebp)
fmull  -0x18(%ebp)
fstpl  -0x10(%ebp)
movl   $0x0,0x4(%esp)
movl   $0x0,(%esp)
call   ba <main+0x48>
fldl   -0x10(%ebp)
fstpl  0x4(%esp)
mov    %eax,(%esp)
call   c9 <main+0x57>
movl   $0x10,0x4(%esp)
movl   $0x0,(%esp)
call   dd <main+0x6b>
mov    $0x0,%eax
add    $0x34,%esp
pop    %ecx
pop    %ebp
lea    -0x4(%ecx),%esp
ret
```

On Tap For Today

- Syllabus
- Programming Languages
- Python vs. C++
- Practice

“Popular” Programming Languages

1. Python
2. JavaScript
3. Java
4. C#
5. C / C++

- https://www.northeastern.edu/graduate/blog/most-popular-programming-languages/

1. JavaScript
2. Python
3. Java
4. C / C++
5. PHP

- https://www.zdnet.com/article/top-programming-languages-most-popular-and-fastest-growing-choices-for-developers/

1. Python
2. Java
3. JavaScript
4. C++
5. C#

- https://www.techrepublic.com/article/the-best-programming-languages-to-learn-in-2022/

1. Python
2. C
3. Java
4. C++
5. C#

- https://www.tiobe.com/tiobe-index/

Accessed Jan 2022

“Common” Language Usages

- Python: Data Science, Machine Learning
- C / C++: Embedded, OS
  - Windows/macOS/Linux kernels written in C & C++
- Java: Mobile, Desktop
- JavaScript: Web, Backend
- Kotlin: Mobile (Android), AR/VR
- Swift: Mobile (iOS), AR/VR

Brief Programming Language History

- C (1972 – Bell Labs)
  - Successor to B to construct utilities on Unix
  - Used for operating systems, embedded systems, supercomputers
- C++ (1985 – Bjarne Stroustrup)
  - Extension of C: “C with Classes”
  - Used for resource-constrained applications and low-level memory manipulation
- Python (1991 – Guido von Rossum)
- Java (1995 – Sun Microsystems)

Comparing Languages

- Python
  - Interpreted
  - Dynamically typed
  - Garbage-collected
  - White space matters

- C++
  - Compiled
  - Statically typed
  - Memory allocation
  - White space doesn't matter

Hello, Python!

```
print("Hello, World!”)
```

```
C:\python.exe helloworld.py

Hello, World!
```

Hello, C++!

```
#include <iostream>


int main() {
   std::cout << "Hello, World!";
   return 0;
}
```

```
C:\g++.exe -o HelloWorld.exe helloworld.cpp

C:\HelloWorld.exe

Hello, World!
```

Python Max Function

```
def max(a, b):
    if(a >= b):
        return a
    else:
        return b

print(f'Max is {max(3,4)}')
```

C++ Max Function v0

```
#include <iostream>
int max(int a, int b)
{
  if(a >= b)
  {
    return a;
  }
  else
  {
    return b;
  }
}
int main()
{
  std::cout << "Max is " << max(3, 4) << std::endl;
  return 0;
}
```

C++ Max Function v1

```
#include <iostream>
int max(int a, int b) {
  if(a >= b) {
    return a;
  } else {
    return b;
  }
}
int main() {
  std::cout << "Max is " << max(3, 4) << std::endl;
  return 0;
}
```

C++ Max Function v2

```
#include <iostream>
int max(int a, int b) {
  if(a >= b)
    return a;
  else
    return b;
}
int main() {
  std::cout << "Max is " << max(3, 4) << std::endl;
  return 0;
}
```

C++ Max Function v3

```
#include <iostream>
int max(int a, int b) {
  if(a >= b) return a;
  else return b;
}
int main() {
  std::cout << "Max is " << max(3, 4) << std::endl;
  return 0;
}
```

C++ Max Function v4

```
#include <iostream>
int max(int a, int b) { if(a >= b) return a; else
return b; }
int main() { std::cout << "Max is " << max(3, 4) <<
std::endl; return 0; }
```

C++ Max Function v5

```cpp
#include <iostream>

int max(int a, int b) {
  return (a >= b ? a : b);
}

int main() {
  std::cout << "Max is " << max(3, 4) << std::endl;
  return 0;
}
```

C++ Max Function v6

```
#include <iostream>
using namespace std;

int max(int a, int b) {
  return (a >= b ? a : b);
}

int main() {
  cout << "Max is " << max(3, 4) << endl;
  return 0;
}
```

C++ Max Function v7

```
#include <iostream>
using namespace std;

const float EPS = 0.000001f;

float max(float a, float b) {
  return (a >= b - EPS ? a : b);
}

int main() {
  cout << "Max is " << max(3.3333333f, 3.3333334f)
       << endl;
  return 0;
}
```

On Tap For Today

- Syllabus
- Programming Languages
- Python vs. C++
- Practice

To Do For Next Time

- Set 0 (due Thursday Jan 11)
  - Get your computer setup (if desired...recommended)
    - Questions: post to Ed Discussion for help
  - Purchase zyBooks (work on Chapter 1)
  - Complete surveys
- Next Time (Friday Jan 12)
  - Come in, have VS Code open & ready to go