

Outline

- 1 Compilation and JVM
- 2 Memory
- 3 Integers
- 4 Adding and Removing Bits
- 5 Bit Operations

Outline

- 1 Compilation and JVM
- 2 Memory
- 3 Integers
- 4 Adding and Removing Bits
- 5 Bit Operations



Hardware

```
01010101
11011011
01110111
```

Machine Code

```
movl %eax, -12(%rbp)
movl %esi, %eax
addq $16, %rsp
```

x86-64 Assembly

```
char *x = malloc(sizeof(char));
*x = 'a';
printf("%c\n", x);
free(x);
```

C Program



Java Virtual Machine

```
64 05 6c
```

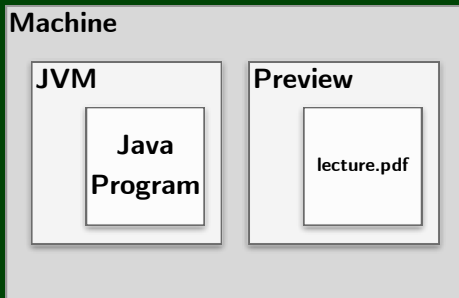
Java Bytecode

```
isub
iconst_2
idiv
```

Java Bytecode
Instructions

```
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello");
    }
}
```

Java Program



Overview

In this project, you will implement all the integer JVM instructions. Your JVM will be able to run **real** compiled class files.

Learning Outcomes

- I can distinguish between how Java and C execute on a computer.
- I can identify the different levels of expressiveness between assembly/bytecode and statements in a high-level programming language.
- I can describe how code can be viewed as a type of data.
- I can write a virtual machine.

Outline

- 1 Compilation and JVM
- 2 Memory
- 3 Integers
- 4 Adding and Removing Bits
- 5 Bit Operations

Memory, Addresses, and Pointers

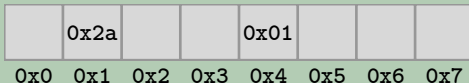
- **Memory** is (essentially) a large array of bytes.
- An **address** is an index into that array.
- A **pointer** is a variable that stores an address.

```
1 char *p = malloc(sizeof(char));  
2 *p = 42;  
3 printf("p = %p\n", p);  
4 printf("*p = %p\n", *p);  
5 printf("&p = %p\n", &p);
```

OUTPUT

```
>> p = 0x01  
>> *p = 0x2a  
>> &p = 0x04
```

A Picture of Memory




```
1 char **p = malloc(sizeof(char *));
2 *p = malloc(sizeof(char));
3 **p = 42;
4 printf("p = %p\n", p);
5 printf("*p = %p\n", *p);
6 printf("**p = %p\n", **p);
7 printf("&p = %p\n", &p);
8 printf("&*p = %p\n", &*p);
9 printf("*&p = %p\n", *&p);
```

OUTPUT

```
>> p = 0x0a
>> *p = 0x04
>> **p = 0x2a
>> &p = 0x09
>> &*p = 0x0a
>> *&p = 0x0a
```

A Picture of Memory

