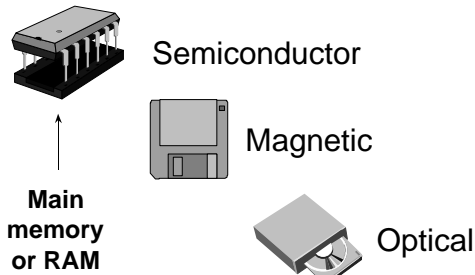


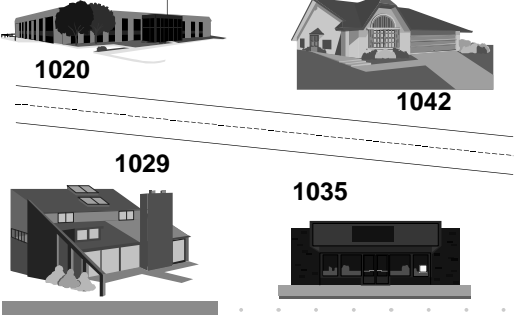
Computer Memory



Semiconductor Memory

- **Millions of devices (transistors)**
 - Each on (1) or off (0)
- **Binary digit (bit)**
 - Symbols 0 .. 1 (like decimal: 0 .. 9)
- **Grouped in octets**
 - One byte = 8 bits

Neighborhood Organization



Neighborhood Organization

Content:

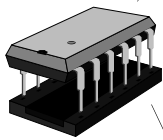
- What is at location
- Value?



1042

Address: identifies location

Memory Organization



| Address | Data (value) |
|---------|--------------|
| 0 | 01101101 |
| 1 | 01110010 |
| 2 | 11000110 |
| ---- | ---- |
| n-1 | 10111000 |

Storing Information

- **Program has data objects**
 - Stored in computer memory
 - Compiler sets aside memory space
- **Data object occupies memory**
 - Some number of bits
 - Varies by data type and compiler

Data Object Characteristics

- **Address**
 - Location in memory
- **Name**
 - For reference in program
- **Type**
 - Size of object
 - Way to interpret
- **Value**
 - Actual bits
 - Interpreted appropriately

Data Object Definitions

- **To create data object**
- **Specify:**
 - Name
 - Type (and const or non-const)
 - Initial value
 - Optional if non-const
 - Often convenient

Definition Examples

```
int count = 0;
const int maxGrade = 100;
float x;
float y = 4.7;
double rocketSpeed;
const int daysInWeek = 7;
char initial = 'A';
const float lightSpeed = 1.863e5;
```

Name
Type (const?)
Initial value

Document all
declarations?
(Text, page 43)

Definition: Alternate Form

```

char (initial = 'A');
const float (lightSpeed = 1.863e5);
char (initial('A'));
const float (lightSpeed(1.863e5));

```

Placement of Definitions

- **Before first use of object**
- **Otherwise, anywhere** (unlike C)
 - May be defined as needed
 - "Just in time"
- **Often group related objects**
 - Sometimes at beginning of section

Identifiers

- **Names of things in a program**
 - Including data objects
- **Some are predefined**
 - By language or libraries you include
- **Naming rules**

Identifier Naming Rules

- **Sequence of characters**
 - Alphabetic (upper or lower case)
 - Case is significant
 - Digits
 - Underscore
- **May not begin with a digit**
- **Not a keyword** (reserved by C++)

Naming Conventions

- **Seek consistency, clarity**
 - address
 - weight
 - count
- **Programmer's choice**
 - student_name
 - course_num
 - total_of_scores
- Or team, organization standard
 - studentName
 - courseNum
 - totalOfScores

Expressions

- **Computation building block**
- **Operands**
 - Data objects (variables, constants)
 - Literal values (e.g., numbers)
- **Operators**
 - E.g., “=” and “+”

⋮

Expression Examples

| | |
|--------------------|----------------|
| count + 3 | Data objects |
| 35 | Literal values |
| 41 - 17.2 | Operators |
| cin >> initial | Side effects |
| x = 1 / lightSpeed | |

.....

⋮

Expression Statements

- **Terminated with semicolon**
–Used primarily for side effects?

```
cin >> initial; ←
x = 1 / lightSpeed; ←
```
- **Null statement**
–Semicolon only: no resulting action

.....

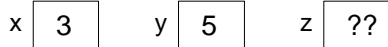
⋮

Assignments x = y + 3;

- **Store a value in a data object**
- **Assignment operator (=)**
- **Lvalue** (memory address)
–Left-hand-side name (variable)
- **Rvalue**
–Right-hand-side value (expression)

Assignment Example (Initialization)

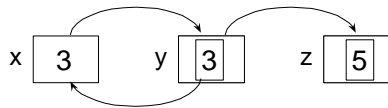
```
int x = 3;
int y = 5;
int z;
```



Assignment Example (Assignment)

```
z = y;
y = x;
x = y;
```

Note: "=" not the same as equality in mathematics!



Illegal Assignments

```
3 = x; // illegal
x + y = 12; // illegal
```

Left side of "=" must be an lvalue, representing a memory address

What Does This Do?

`x = x + 8 ;`

(2) Store the result in this data object

(1) Compute the value of this expression

Another form: `x += 8 ;`
