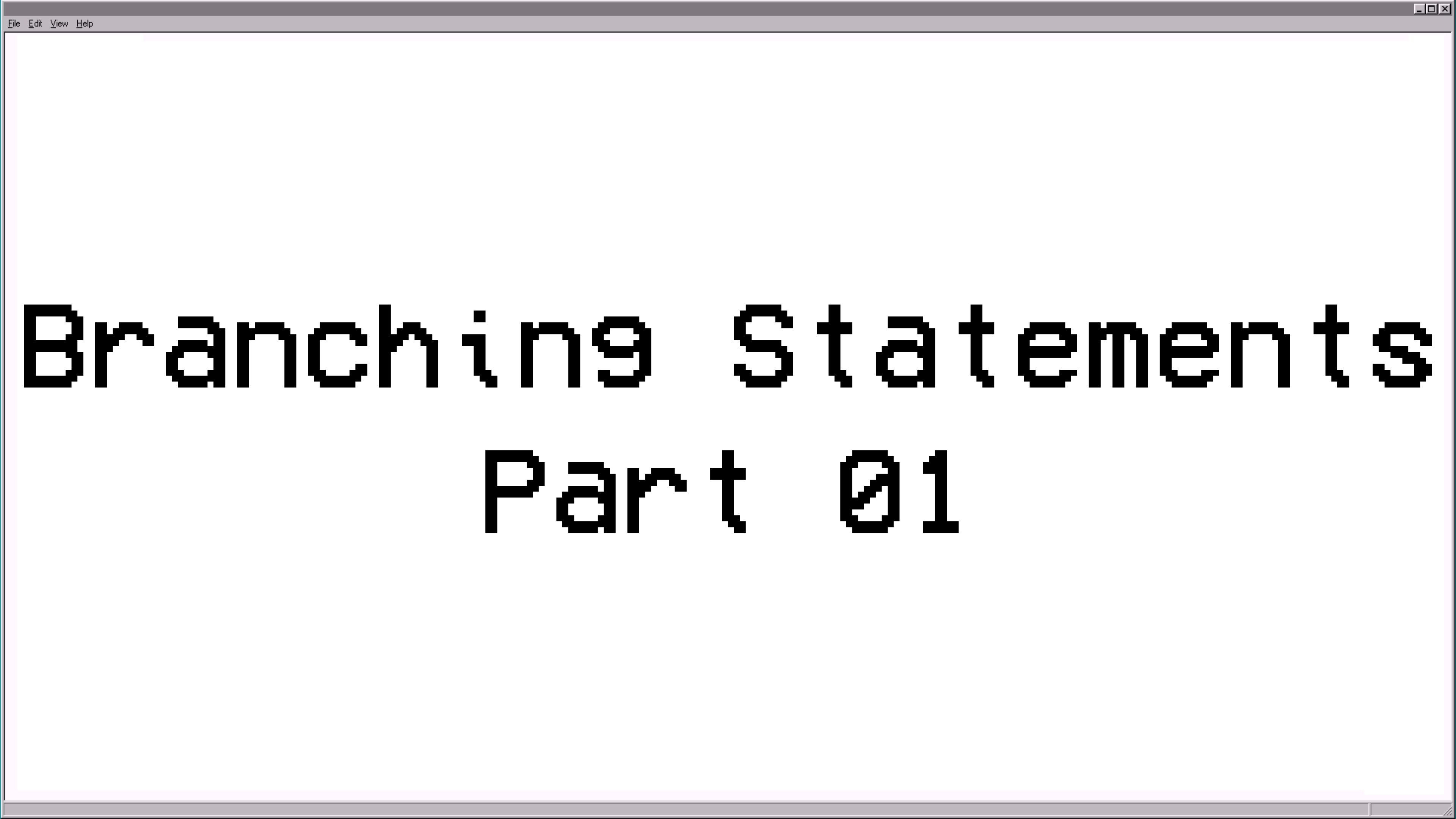
File
Edit
View
Help



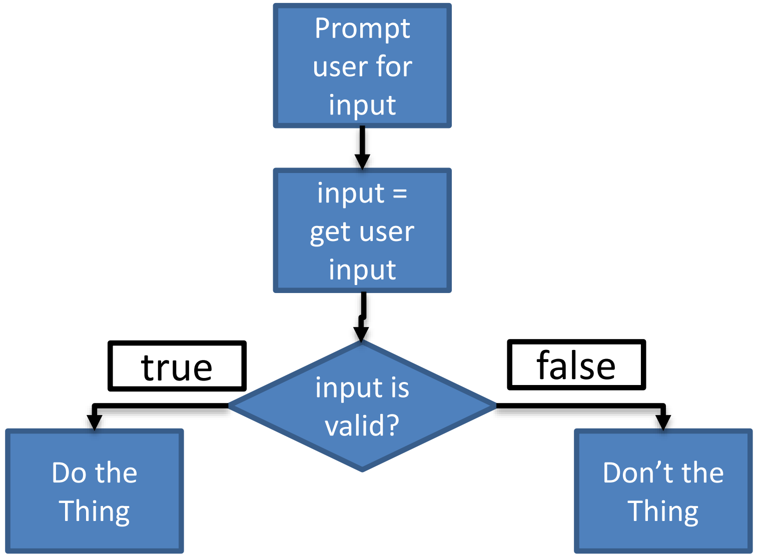
Branching Statements
Part 01

File Edit View Help

Flow Control

- Flow of control is the order in which a program performs actions.
- A **branching statement** chooses between two or more possible actions.
- A **loop statement** repeats an action until a stopping condition occurs.
- Flow Charts diagram the flow of a program
  - Boxes are Statements
  - Diamonds are Decisions
    - True branch
    - False branch
  - Arrows indicate the flow of statements and decisions
  - Pseudocode is mostly used

Flow Chart Example



Branching Statements

- If-statement
- If the Boolean expression is "true" then the body of the if-statement is executed, and otherwise is ignored
- Putting curly braces "{}" to denote the body of the if-statement is strongly encouraged
- Do not put a semicolon ";" after the parenthesis
  - It will ignore the Boolean expression
- Spoken
  - "if this is true then do this"

Syntax

```
if(<<Boolean expression>>)
{
	//Body of the if-statement
}
//Outside Body of the if-statement
```

Examples

```
if(a == b)
{
	System.out.println("a is equal to b");
}
```

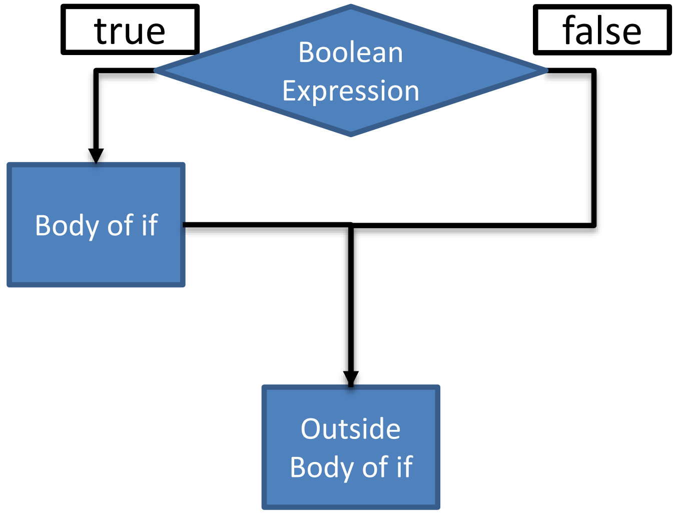
File Edit View Help

Branching Statements

Syntax

```
if(<<Boolean expression>>)
{
    //Body of the if-statement
}
//Outside Body of the if-statement
```

General If-Statement Flow Chart



File Edit View Help

Branching Statements

- Else-statement
- Requires a proceeding if-statement
  - If-statements do not require an else-statement
- If the Boolean expression is "false" then the body of the else-statement is executed, and otherwise is ignored
- Putting curly braces "{}" to denote the body of the else-statement is strongly encouraged
- Spoken:
  - "if this is true then do this, otherwise (else) do that"

Syntax

```
if(<<Boolean expression>>)
{
	//Body of the if-statement
}
else
{
	//Body of the else-statement
}
```

Examples

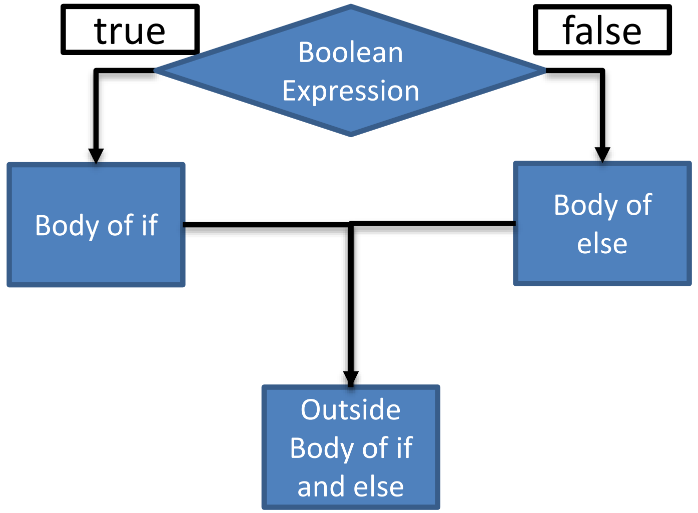
```
if(a == b)
{
	System.out.println("a is equal to b");
}
else
{
	System.out.println("a is not equal to b");
}
```

Branching Statements

Syntax

```
if(<<Boolean expression>>)
{
    //Body of the if-statement
}
else
{
    //Body of the else-statement
}
//Outside of if and else
```

General If-Else-Statement Flow Chart



File Edit View Help

Boolean Expressions

- True or False Value
- Common Boolean Operators
  - “==“ : Equal to
  - “!=“ : Not Equal
  - “<“ : strictly less than
  - “>” : strictly greater than
  - “<=“: less than or equal to
  - “>=“: greater than or equal to

<u>Syntax</u>

```
<<value>> <<Boolean operator>> <<value>>;
```

<u>Examples</u>

```
boolean a = 12 > 3;
if(a)//Or a == true
{
        System.out.println(“Here”);
}
else
{
        System.out.println(“Not here”);
}
```

File Edit View Help

Compound Boolean Expressions

- Combines multiple Boolean expressions
- Common Compound Boolean Expression Operators
  - "&&" : AND – both must be true to yield true
  - "||" : OR – only one must be true to yield true

Syntax

```
<<Boolean expression>> <<operator>> <<Boolean expression>>;
```

Examples

```
boolean a = 2 != 0 && 12 > 3;
if(a)//Or a == true
{
        System.out.println("Here");
}
else
{
        System.out.println("Not here");
}
```

File Edit View Help

Compound Boolean Expressions

Truth Table

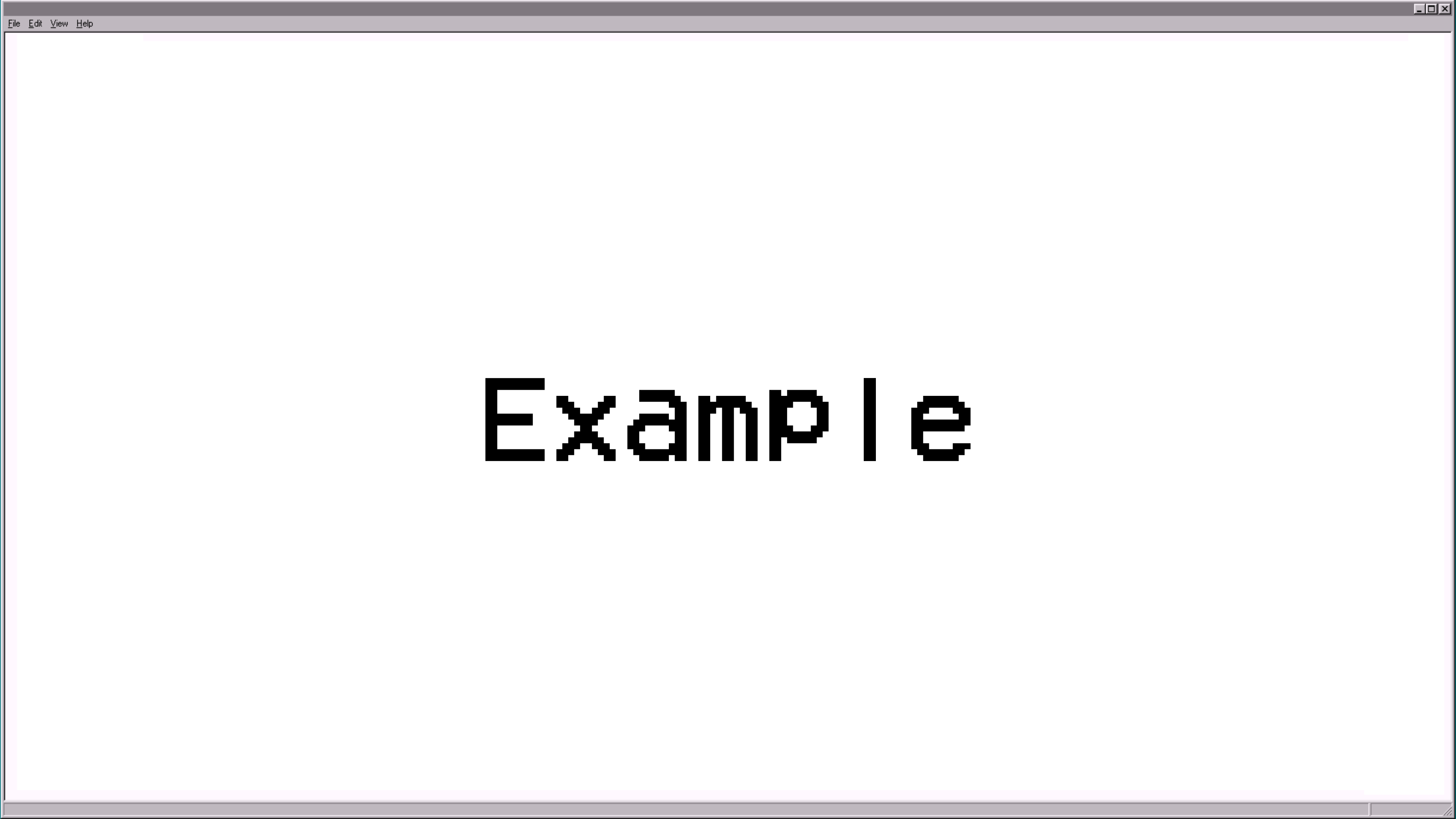
A	B	A && B	A || B
TRUE	TRUE	TRUE	TRUE
TRUE	FALSE	FALSE	TRUE
FALSE	TRUE	FALSE	TRUE
FALSE	FALSE	FALSE	FALSE

File Edit View Help

Negating Boolean Expressions

- The NOT operator “!” is used to negate the value of a Boolean expression

Negated Expression	Equivalent Expression
!(A < B)	(A >= B)
!(A <= B)	(A > B)
!(A > B)	(A <= B)
!(A >= B)	(A < B)
!(A == B)	(A != B)
!(A != B)	(A == B)
!(A && B)	(A || B)
!(A || B)	(A && B)



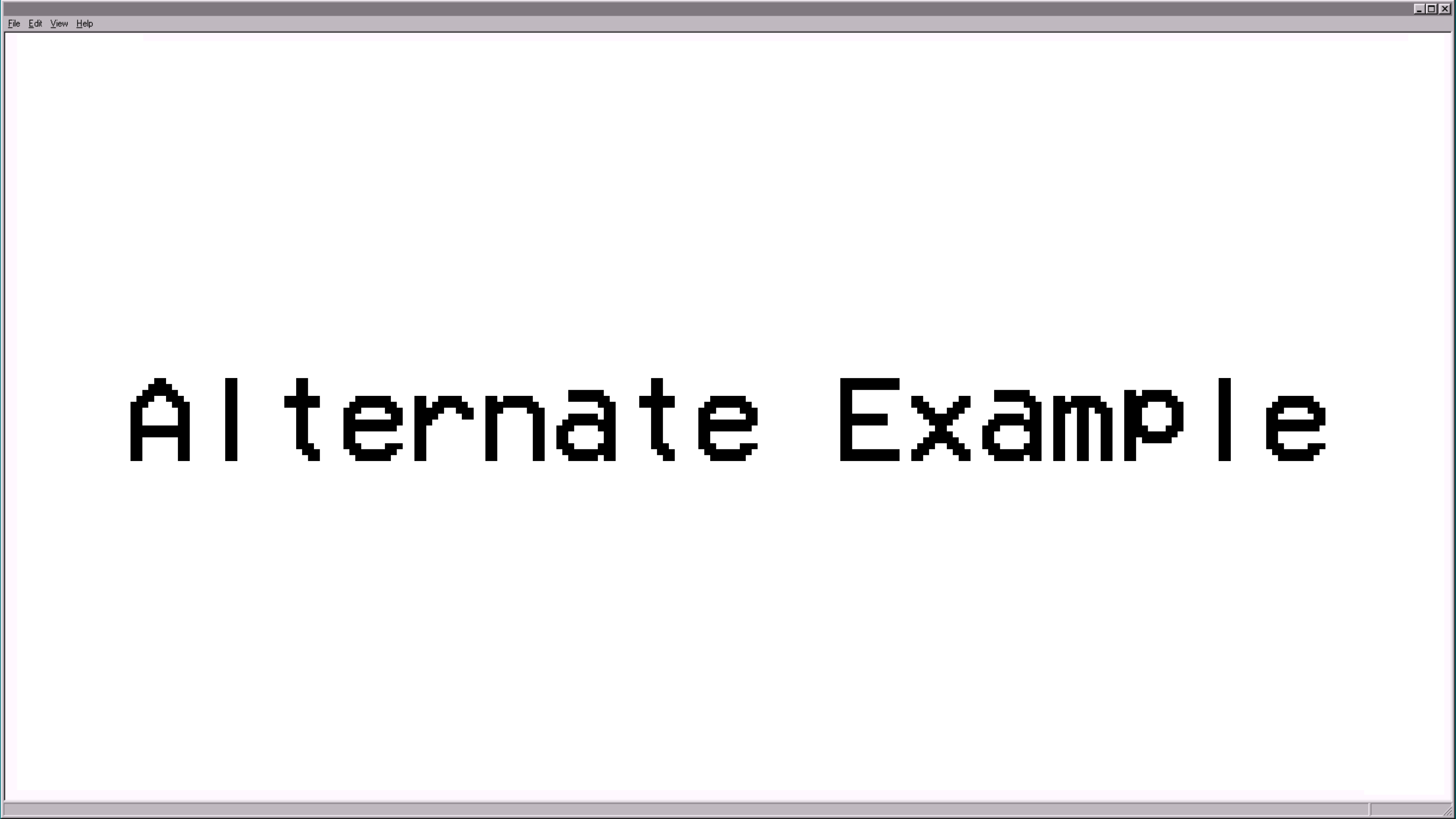
Example

File Edit View Help

Using "=="

- The operator "==" is great for determining if two values are equal in some cases, but not all
- Great to use when comparing integer values
- Not great to use when comparing floating-point values
  - Round of errors
  - Use a combination of >= and <= with a tolerance

- Great for comparing memory addresses of Objects
  - Check if objects are NULL
  - Check if two identifiers reference the same place in memory
- Not great for comparing contents of an object
  - Use the ".equals()" method instead



Alternate Example