Representations

- What you should know:
  - Base$^{(\#\ digits)}$ = number of combinations
  - Examples:
    - Decimal:
      1 decimal digit = 10 combinations (0-9)
      2 decimal digits = 100 combinations (0-99)
      Etc.
    - Binary
      1 binary digit = 2 combinations (0 to 1)
      2 binary digits = 4 combinations (00 to 11)
      Etc.
    - N^{ary} base e.g. n = 8 (octal), 16 (hexadecimal) etc.
      1 digit = N combinations (0 to N-1)
      2 digits = N x N combinations (0 to (NxN)-1)
      3 digits = N x N x N combinations (0 to (NxNxN)-1)

James Tam

Applying Precedence/Order Of Operation

- **Name of the complete example:** `1determingOrder.py`

James Tam

'Classic' Questions That Apply These Concepts

- Generic question:
  - Given you have 9 values to represent what is the minimum number of bits needed to store all n values.

- Tam question: The Borg[1] alphabet consists 66 symbols. How many bits would the Bynars[1] (0 to (NxN)-1) need to encode each letter?

- Try to work these out on your own (answer at the end of these notes).

1 © Paramount

James Tam

Applying Precedence/Order Of Operation

- How to evaluate a logical expression when bracketing is not used.[1]
- Example:

```
if True or False and not True:
    print("Expression true")
else:
    print("Expression false")
```

1 JT's comment: you should always use brackets to make things clear but you may be required to trace the code of another person does not.

James Tam

Applying Precedence/Order Of Operation

- How to evaluate a logical expression when bracketing is not used.
- Evaluating the Boolean expression:

```
True or False and not True
```

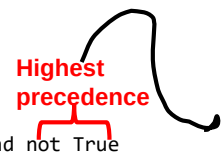
Precedence rules

Order	Operations	Precedenc
1	()	Highest
2	x ** y	
3	-x, +x	
4	x * y, x / y, x % y, x // y	
5	x + y, x - y	
6	<, <=, >, >=	
7	!=, ==	
8	not	
9	and	
10	or	
11	=	Lowest

James Tam

Applying Precedence/Order Of Operation

- How to evaluate a logical expression when bracketing is not used.
- Evaluating the Boolean expression:




```
True or False and not True
```

Precedence rules

Order	Operations	Precedenc
1	()	Highest
2	x ** y	
3	-x, +x	
4	x * y, x / y, x % y, x // y	
5	x + y, x - y	
6	<, <=, >, >=	
7	!=, ==	
8	not	
9	and	
10	or	
11	=	Lowest

James Tam

Applying Precedence/Order Of Operation

- How to evaluate a logical expression when bracketing is not used.
- Evaluating the Boolean expression:
  - NOT has been evaluated

Precedence rules

Order	Operations	Precedenc
1	()	Highest
2	x ** y	
3	-x, +x	
4	x * y, x / y, x % y, x // y	
5	x + y, x - y	
6	<, <=, >, >=	
7	!=, ==	
8	not	
9	and	
10	or	
11	=	Lowest

James Tam

Applying Precedence/Order Of Operation

- How to evaluate a logical expression when bracketing is not used.
- Evaluating the Boolean expression:
  - AND has been evaluated

Precedence rules

Order	Operations	Precedenc
1	()	Highest
2	x ** y	
3	-x, +x	
4	x * y, x / y, x % y, x // y	
5	x + y, x - y	
6	<, <=, >, >=	
7	!=, ==	
8	not	
9	and	
10	or	
11	=	Lowest

James Tam

Applying Precedence/Order Of Operation

- How to evaluate a logical expression when bracketing is not used.
- Evaluating the Boolean expression:
  - OR has been evaluated
  - Final result of compound Boolean

```
if(True):
    print("Expression true")
```

Precedence rules

Order	Operations	Precedenc
1	()	Highest
2	x ** y	
3	-x, +x	
4	x * y, x / y, x % y, x // y	
5	x + y, x - y	
6	<, <=, >, >=	
7	!=, ==	
8	not	
9	and	
10	or	
11	=	Lowest

James Tam

Precedence/Order Of Operation: A More Complex Example

- **Name of the complete example:** `2determingOrder.py`

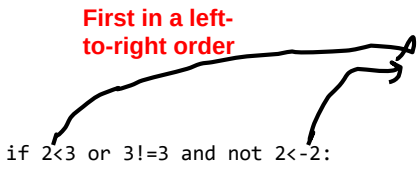
James Tam

Applying Precedence/Order Of Operation

- How to evaluate a logical expression when bracketing is not used.
- Evaluating the Boolean expression in the branch.

Precedence rules

Order	Operations	Precedenc
1	()	Highest
2	x ** y	
3	-x, +x	
4	x * y, x / y, x % y, x // y	
5	x + y, x - y	
6	<, <=, >, >=	
7	!=, ==	
8	not	
9	and	
10	or	
11	=	Lowest




```
if 2<3 or 3!=3 and not 2<-2:
```

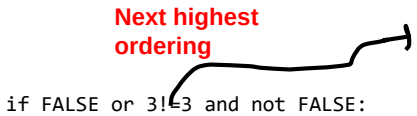
James Tam

Applying Precedence/Order Of Operation

- How to evaluate a logical expression when bracketing is not used.
- Evaluating the Boolean expression in the branch.

Precedence rules

Order	Operations	Precedenc
1	()	Highest
2	x ** y	
3	-x, +x	
4	x * y, x / y, x % y, x // y	
5	x + y, x - y	
6	<, <=, >, >=	
7	!=, ==	
8	not	
9	and	
10	or	
11	=	Lowest




```
if FALSE or 3!=3 and not FALSE:
```

James Tam

Applying Precedence/Order Of Operation

- How to evaluate a logical expression when bracketing is not used.
- Evaluating the Boolean expression in the branch.

Precedence rules

Order	Operations	Precedenc
1	()	Highest
2	x ** y	
3	-x, +x	
4	x * y, x / y, x % y, x // y	
5	x + y, x - y	
6	<, <=, >, >=	
7	!=, ==	
8	not	
9	and	
10	or	
11	=	Lowest

Next highest ordering

```
if FALSE or TRUE and not FALSE:
```

James Tam

Applying Precedence/Order Of Operation

- How to evaluate a logical expression when bracketing is not used.
- Evaluating the Boolean expression in the branch.

Precedence rules

Order	Operations	Precedenc
1	()	Highest
2	x ** y	
3	-x, +x	
4	x * y, x / y, x % y, x // y	
5	x + y, x - y	
6	<, <=, >, >=	
7	!=, ==	
8	not	
9	and	
10	or	
11	=	Lowest

Next highest ordering

```
if FALSE or TRUE and TRUE:
```

James Tam

Applying Precedence/Order Of Operation

- How to evaluate a logical expression when bracketing is not used.
- Evaluating the Boolean expression in the branch.

Precedence rules

Order	Operations	Precedenc
1	()	Highest
2	x ** y	
3	-x, +x	
4	x * y, x / y, x % y, x // y	
5	x + y, x - y	
6	<, <=, >, >=	
7	!=, ==	
8	not	
9	and	
10	or	
11	=	Lowest

Remaining operator

```
if FALSE or TRUE:
```

James Tam

Applying Precedence/Order Of Operation

- How to evaluate a logical expression when bracketing is not used.
- Evaluating the Boolean expression in the branch.

Precedence rules

Order	Operations	Precedenc
1	()	Highest
2	x ** y	
3	-x, +x	
4	x * y, x / y, x % y, x // y	
5	x + y, x - y	
6	<, <=, >, >=	
7	!=, ==	
8	not	
9	and	
10	or	
11	=	Lowest

Final result

```
if TRUE:
```

The Tamman says again: "Bracket everything".
You will sleep better and live a happier more fulfilling life (maybe not but you'll aggravate your programming team mates less).

James Tam

Answer To The Questions Applying Representations

James Tam

'Classic' Questions That Apply These Concepts

- **Generic question:**
  - Given you have 9 values to represent what is the minimum number of bits needed to store all n values.
  - **Answer: 4 bits**. Using 3 bits is insufficient as not all values can be store with only 8 combinations. Although using 4 bits results in 7 used combinations this occurs in real life applications e.g. ASCII only requires 7 bits to encode the character information but 8 are used.

- **Tam question:** The Borg[1] alphabet consists 66 symbols. How many bits would the Bynars[1] (0 to (NxN)-1) need to encode each letter?
  - **Answer: 7 bits.**

1 © Paramount

James Tam