Snobol
and
Snowflake

COMP360

*"The original name was **S**ymbolic **EX**pression **I**nterpreter (SEXI)*

It was clear that we needed another name!! We sat and talked and drank coffee and shot rubber bands and after much too much time someone said "We don't have a Snowball chance in hell of finding a name". All of us yelled at once, "WE GOT IT – SNOBOL" in the spirit of all the BOL languages. We then stretched our mind to find what it stood for."

David Farber

Schedule

- The lexical scanner for the Snowflake language is due by midnight on Wednesday, February 12, 2020
- The first exam is Friday, February 14, 2020

And now for something completely different

- In this class we will be creating the first two (maybe three) phases of a compiler for a weird, different, silly, amazing *(pick one)* made up language called Snowflake
- Snowflake is a subset of Snobol
- The Snowflake language does pattern matching and string manipulation
- For simplicity, Snowflake does not do any arithmetic, input, output or functions

Different Language Style

- Snobol is significantly different from most languages
- It is an older procedural language
- Snobol is designed for string manipulation
- Pattern matching is a basic part of Snobol

- Snowflake, the toy language we will be compiling, is similar to Snobol, but much simpler

Snobol

- SNOBOL (**S**tri**N**g **O**riented and sym**BO**lic **L**anguage) is a series of computer programming languages developed between 1962 and 1967 at AT&T Bell Laboratories by David J. Farber, Ralph E. Griswold and Ivan P. Polonsky
- Snobol only has one statement type
- Snobol is line oriented, probably because it was created in the days of punch cards

Simple Stuff

- Snobol does not require variables to be declared
- Strings and numbers can be used together (like Python)

```
cat = 'dog'
bird = '17' + 3.21
```

- *Note: Snowflake does not have numbers*

Concatenation

- The default Snobol operation is concatenation
- If two variables or constants are adjacent, they are concatenated

```
eagle  = 'big'
canary = eagle 'bird'
```

- canary has the value `'bigbird'`

Input and Output

Snobol has two special variables

- **output** – when you assign a value to the variable output, the value is written to the screen
- **input** – when you use the variable input, it reads a line from the keyboard

- output cannot be used on the right side of an equals sign and input cannot be on the left

Example using input and output

```
OUTPUT = 'What is your name?'
Username = INPUT
OUTPUT = 'Thank you, ' Username
END
```

Pattern Matching

- You can search one string to see if it contains another

variable pattern = replacement

- The equals sign and replacement are optional in Snobol

```
dog = 'Computer Tech'
dog 'Tech' = 'Science'
```

- dog has the value 'Computer Science'

Complex Pattern Matching

- Snobol allows complex pattern matching
- A pattern can have concatenation and alternation

```
canary = 'big bird sings'
robin  = 'small dog growls'

canary ('small' | 'big') ' ' ('bird' | 'dog')
robin  ('small' | 'big') ' ' ('bird' | 'dog')
```

- Both strings would match

Replacement

- If a pattern succeeds in finding a match, then that portion of the string can be replaced with another value
- If the pattern fails, there is no replacement

```
canary = 'big bird sings'
canary ('bird' | 'dog') = 'cat'
```

- canary will be `'big cat sings'`

Which Snobol Statement replaces the first occurrence of mouse in cat with rat?

A. rat = cat mouse
B. cat mouse = rat
C. cat = cat (rat | mouse)
D. cat (rat | mouse)

Success or Failure

- A Snobol statement can succeed or fail
  - succeed – pattern matches
  - fail – pattern does not match
  - fail – end of file on input

Flow of Control

- Snobol is an old fashion “GOTO” style language
- All Snobol statements can have a label at the beginning
- The label END is required at the end for Snobol *(but not Snowflake)*
- At the end of a Snobol statement there can be a colon then a label specifying the next line to be executed
- The next line can differ depending on the success or failure of the line

What does this program display?

```
    aardvark = 'ants in your pants'
    anteater 'ant' = 'egg'
    output = anteater
end
```

A.ant
B.egg
C.eggs in your pants
D.eggs in your peggs
E.none of the above

Jumping

- An unconditional jump is

```
dog = 'cat'                    :(rabbit)
```

- To jump only if there is a success

```
dog 'cat'                      :s(rabbit)
```

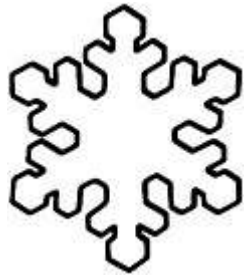
- To jump only if there is a failure

```
dog 'cat'                      :f(rabbit)
```

- To jump either way

```
dog 'cat'                 :s(rabbit)f(hare)
```

Snowflake Simplification



- Snowflake does not use the :(label) style jumps
- Snowflake has a while statement (which Snobol does not)

```
while variable pattern {
        # loop stuff
}
```

- If the pattern is found in the variable, the loop is executed

What does this program display?

```
     aardvark = 'ants in your pants'
zot  anteater 'ant' = 'egg'          :s(zot)
     output = anteater
end
```

A.ant
B.egg
C.eggs in your pants
D.eggs in your peggs
E.none of the above

Using Patterns

```
      OUTPUT = 'What is your name?'
      Username = INPUT
      Username 'J'                                  :S(LOVE)
      Username 'K'                                  :S(HATE)
      OUTPUT = 'Hi, ' Username                      :(END)
LOVE  OUTPUT = 'Nice to meet you, ' Username        :(END)
HATE  OUTPUT = "Oh. It's you, " Username
END
```

Snobol Functions

Snobol has several function that can be used in patterns or expressions

- break('xyz') – match until you encounter any of these characters
- span('xyz') – match a sequence of any of these characters
- trim(' xyz ') – remove leading and trailing spaces
- gt('a', 'b') – succeeds if a is greater than b

- *Snowflake does not have functions*

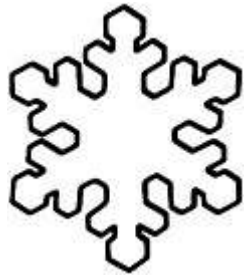
Saving Pattern Matchings

- The value that matches in a pattern can be saved in a variable by putting a $ and the variable name after the pattern

```
canary = 'big bird sings'
canary ('bird' | 'dog') $ cow = 'cat'
```

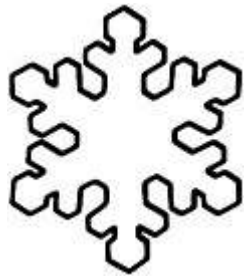
- canary will be `'big cat sings'` and cow will have the value `'bird'`

Snowflake



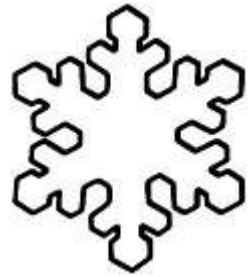
- This semester we will create a compiler for Snowflake
- The Scanner assignment has been posted on Blackboard
- There will be a Parsing assignment that uses the output of the Scanner
- We will consider our Snowflake program to be a method called by another program
  - This avoids input and output

Variables



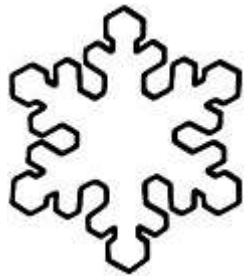
- All Snowflake variables are Strings
- Variables do not have to be declared in Snobol
- Variable names and keywords are case INSENSITIVE
- dog is the same as Dog or DOG or doG
- The case of string constants must be preserved

Semicolons



- Unlike Snobol, Snowflake statements end with a semicolon
- Whitespace is ignored in Snowflake (except in quote strings)

Comments



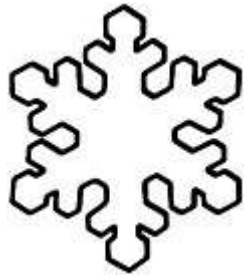
- Comments in Snowflake are in two formats

```
# comment until the end of the line
```

or

```
#!  comment until
      it ends with !#
```

Snowflake Start and End



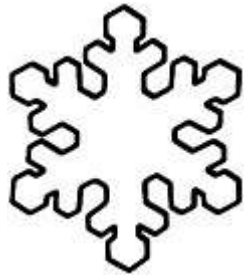
- The first Snowflake statement must be **parm** to define the variables passed to the method as parameters

```
parm dog    cat;
```

- The last Snowflake statement must be **return** to return a value from the method

```
return variable;
```

Assignment



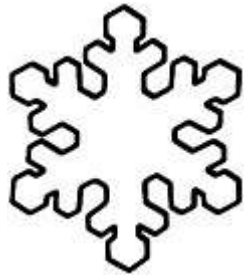
- Simple assignment is like most languages

```
dog = cat; or dog = 'Quote String';
```

- If two or more strings are on the right side of the equals sign, they are concatenated

```
dog = cat 'bird' cow;
```

Patterns

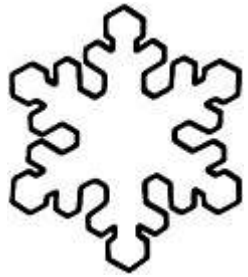


- A pattern is given to the right of a variable on the left of an equals sign

```
dog 'Constant pattern' = cow;
```

- If the pattern exists in the variable dog, it is replaced with the contents of cow
- If the pattern does not exist in the variable dog, nothing happens
- Patterns can be variables or string constants

Multiple Patterns

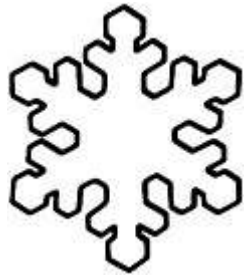


- Patterns can have multiple values to match

```
dog  cat | 'mouse' = rat;
```

- If the value of cat appears in dog, then that portion of the dog string is replaced by the value of rat
- If cat does not appear in dog, then dog is searched to see if “mouse” appears and, if so, it will be replaced by the contents of rat
- There can be many possible pattern options

while Loops

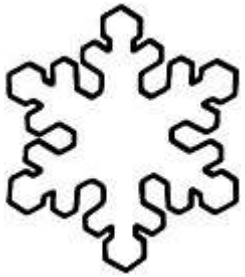


- While loops repeat if a pattern matches

```
while variable pattern {   … }
```

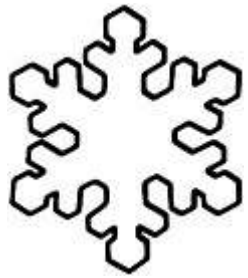
- Patterns work the same way as assignments, but they do not change the value of the variable
- If the pattern exists in the variable (a match success), the loop is executed

Example Snowflake Program



```
parm  dog  cat;
rat = 'gerbil ' cat;
dog cat | 'mouse' = rat;
return dog;
```

What does this Snowflake Program return?



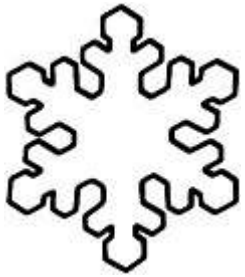
If goat is 'Computer Science' and

cow is 'e'

```
parm  goat  cow;
star = '*';
while goat cow {
    goat cow = star;
    star = star '*';
}
return goat;
```

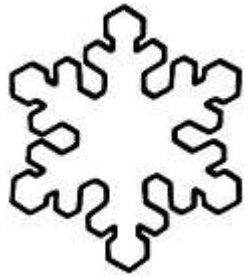
A. *
B. Computer Science*
C. Comput*r Sci*nc*
D. Comput*r Sci**nc***
E. None of the above

Snowflake Summary



- Adjacent variable or strings on the right side of an equals sign are concatenated
- There may be a pattern to the right of the variable on the left of the equals sign
- Patterns may have multiple targets separated by a vertical bar (|)
- Statements end with a semicolon
- Variables are case insensitive

Snowflake Summary



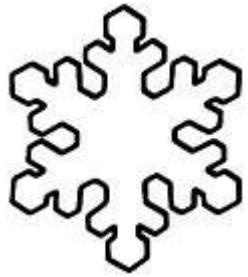
- Comments either

Start with a hash or pound sign to end of line

#! Start with hash exclamation point and end with !#

- String constants start and end with a single 'quote'

Snowflake Summary



- The punctuation used in Snowflake includes:

```
= | { } ;     '
```

- The only keywords are:

```
parm
return
while
```

And now back to our regularly scheduled Programming Languages

Write a BNF for

```
#! sample Snowflake program !#
parm var1  var2  var3 ;
variable = var1  var2;
return variable;
```

Possible Solution

program → parmstmt assign rtn

parmstmt → parm varList **;**

rtn → **return** *variable* **;**

varList → *variable* | *variable* varList

assign → *variable* **=** varList **;**

Schedule

- The lexical scanner for the Snowflake language is due by midnight on Wednesday, February 12, 2020
- The first exam is Friday, February 14, 2020