Lab 2

Quick guide to referencing...
Algorithms, pseudo-code and flow charts

References

Many Acceptable Styles...
APA style Author, X. X. (xxxx). Title of book. Location: Publisher. (source: http://coun.uvic.ca/learning/essays/apa-referencing.html)
IEEE style J. K. Author, “Title of report,” Abbrev. Name of Co., City of Co., Abbrev. State, Rep. *xxx*, year. (source: http://www.ieee.org/documents/ieeecitationref.pdf)
Tips: For online sources: Give URL, and date accessed For material you learned elsewhere: it is ok to say “I learned this in CSc 110 last year...” More info available at: http://library.uvic.ca/instruction/cite/styleguides.html or in person at the library

Plagiarism

http://www.quickmeme.com/meme/3orps3/

References

Some interesting (unacceptable) reference styles :)

Reference that is too vague. For example:

- Google.com
- Wikipedia
- The library

Web source, without URL:

- “Microsoft clipart page”
- “so and so’s home page”

Valid websource, but too vague

- http://www.uvic.ca
- http://www.nytimes.com/

Ask yourself:
If someone gave me this reference, could I use it to *easily* find the information?

Algorithms, Pseudocode and Flowcharts

- Algorithm **solves a problem** if and only if:
  - on any input, it halts
  - on any input, it gives the correct output

Algorithms, Pseudocode and Flowcharts

- Algorithm representation:
  - pseudo code
  - flow chart
  - code (C++, Java, Fortran, BASIC...)
  - ...

Pseudo code

- A text-based representation of the algorithm
- Doesn't conform to a specific syntax
- Still needs to be:
  - unambiguous
  - clear

Pseudo Code Notes

- **return** exits:
- **IF/ELSE/THEN**
  - conditional branching
- **WHILE, FOR, DO WHILE...**
- **INPUT** and **OUTPUT**
- to "call" it, need to name it
- indicate BOUNDS or SCOPE using:
  - indentation or brackets

Pseudo Code Example: Adding 2 bits

- What are the input(s)?
- What are the output(s)?

Pseudo Code Example: Adding 2 bits

- What are the input(s)?
  - ? bits (0 or 1)....
- What are the output(s)?
  - ? bits (0 or 1)....

Pseudo Code Example: Adding 2 bits

- What are the input(s)?
  - 2 bits (0 or 1)....
- What are the output(s)?
  - 2 bits (0 or 1)....
- Assume that "addition" is not already understood

Pseudo Code Example: Adding 2 bits

- ...

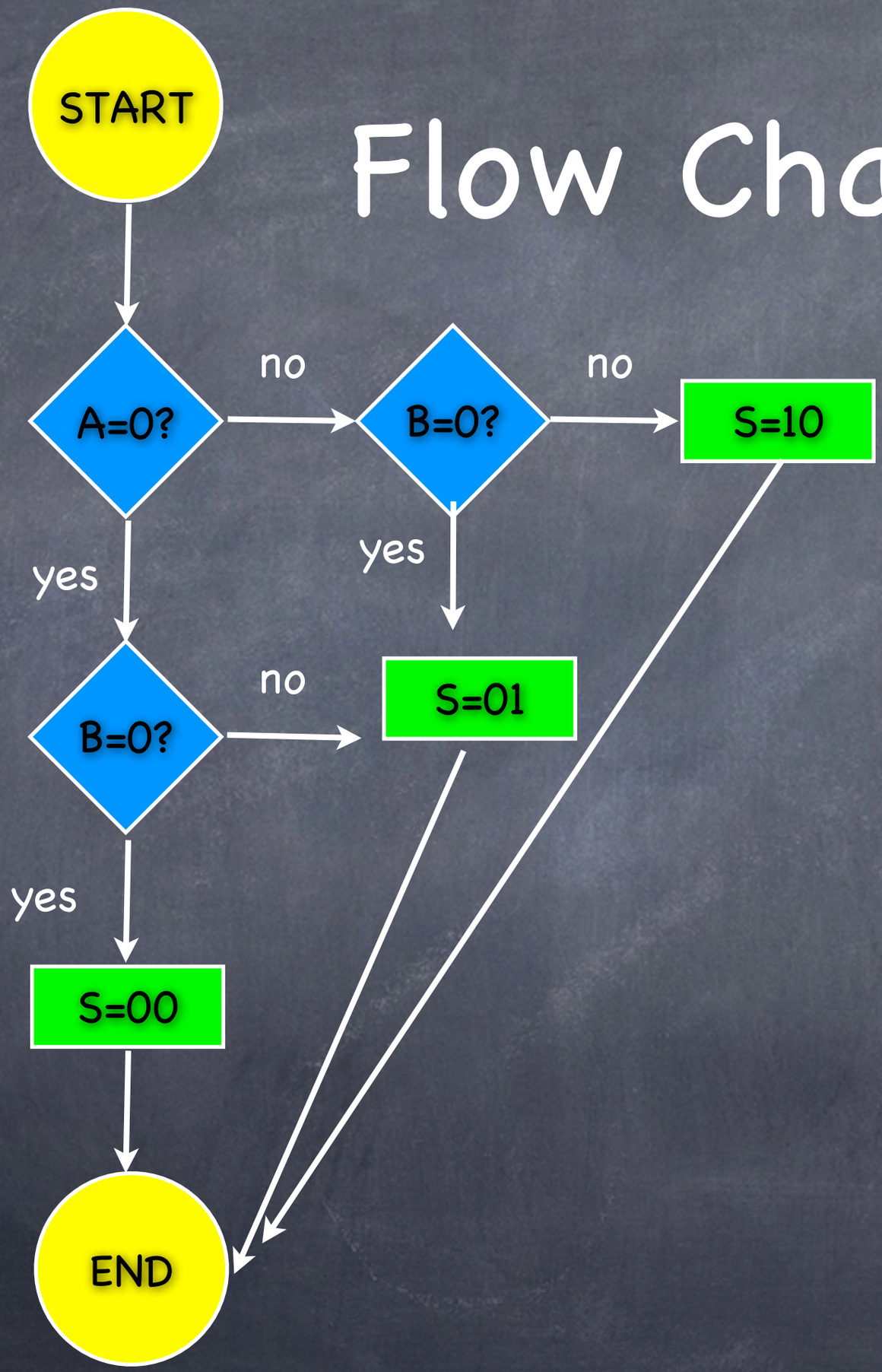
Pseudo Code Example: Adding 2 bits

TWOBITADD

INPUT: A, B (one bit each), OUTPUT: S (2 bits)

```
IF A equals 0 {
IF B equals 0 {
return 00
}ELSE IF B equals 1 {
return 01
}
} ELSE IF A equals 1 {
IF B equals 0 {
return 01
} ELSE IF B equals 1 {
return 10
}
}
```

Flow Chart



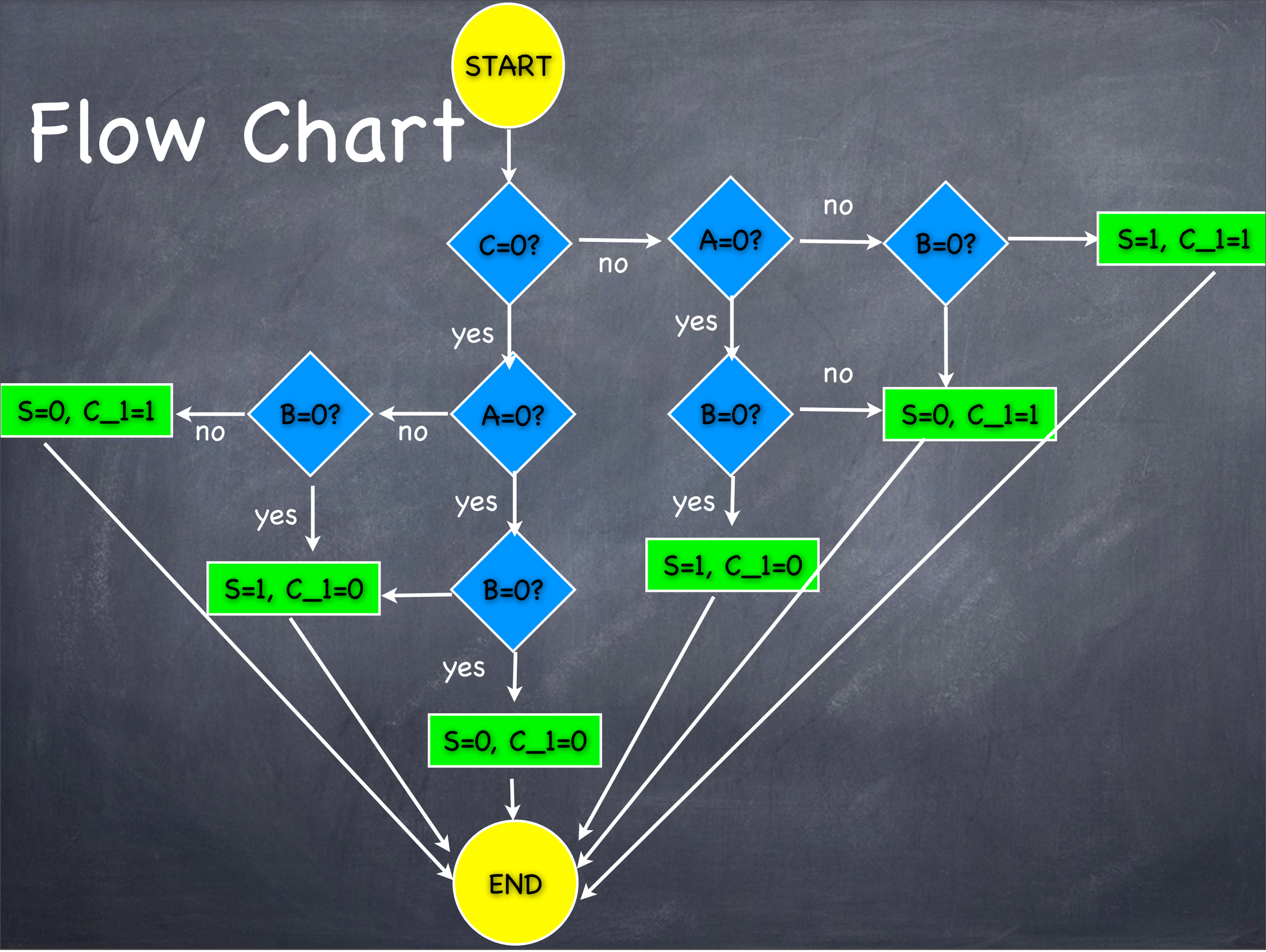
Pseudo Code Example: Adding 2 bits with CARRY

- What are the input(s)?
  - 3 bits (0 or 1)....
- What are the output(s)?
  - 2 bits (0 or 1)....

Pseudo Code Example: Adding 2 bits

```
TWOBITADD_C
INPUT:  A, B, C (one bit each),
OUTPUT: S, C_1 (1 bit each)
 IF C equals 0
     IF A equals 0
     IF B equals 0
          return S=0, C_1=0
     ELSE IF B equals 1
          return S=1, C_1=0
     ENDIF
ELSE IF A equals 1
     IF B equals 0
          return S=1, C_1=0
     ELSE IF B equals 1
          return S=0, C_1=1
     ENDIF
ELSE IF C equals 1
     IF A equals 0
          IF B equals 0
               return S=1, C_1=0
          ELSE IF B equals 1
               return S=0, C_1=1
          ENDIF
     ELSE IF A equals 1
          IF B equals 10
               return S=1, C_1=0
          ELSE IF B equals 1
          return S=1, C_1=1
     ENDIF
ENDIF
```

Flow Chart


START
C=0?
no
A=0?
no
B=0?
S=1, C_1=1
yes
yes
no
S=0, C_1=1
B=0?
no
A=0?
B=0?
yes
yes
yes
S=1, C_1=0
S=1, C_1=0
B=0?
yes
S=0, C_1=0
END

Challenge: Full Binary Addition

- write pseudo code to:
  - add two arbitrarily long binary numbers
  - use the TWOBITADD_C pseudo code
  - call it like TWOBITADD_C(A=?, B=?, C=?)
  - imagine it prints the answer right to left...

Full Two-bit add

Assumptions:
input strings are of equal length (shorter string padded with 0s in highest-order bits)
print function takes a variable and outputs its value

FULLTWOBITADD
INPUT: A, B (binary strings)
OUTPUT: prints out sum

```
let c = 0
WHILE there are bits left in A and B{
let a equal rightmost bit of A
let b equal rightmost bit of B
TWOBITADD_C(a, b, c)
print S
remove rightmost bit of A
remove rightmost bit of B
let c equal value of C_1
}
if c equals 1{
print c
}
```