CS460 & CS210

Manju Muralidharan

PSEUDOCODE CONVENTIONS

This slide deck will serve as a reference to write "good" pseudocode.

By no means is this the only way to write pseudocode but it is a great starting point and will standardize the expectations for exams and assignments for the purpose of the class.

INDENTATION

Always use clear indentation in your psuedocode.

Have clear whitespaces and line breaks where necessary.

Use descriptive terms like “integer,” “string,” or “list” when necessary.

Follow a standard naming conventions

```
returntype functionName(parameter list){
  datatype variable_name = value
}
```

ACTION WORDS

Common Action Keywords

Several keywords are often used to indicate common input, output, and processing operations.

Input: READ, GET
Output: PRINT, DISPLAY, SHOW
Initialize: SET, INIT
Add one: INCREMENT, ADD

IF-THEN-ELSE

```
if condition then {
  //if actions here
}
else{
  //else actions here
}
```

No need to add endif statements. Braces will suffice.

ITERATION AND LOOPING

- Clearly define the start and end conditions for loops.
- Indicate how loop variables are updated within the loop.

```
while (employee.type != manager && personCount < numEmployees){
  increment personCount
}


for i = 1 to 10 {
// for exits when i=11
  print i
  i++
}
}
```

EXAMPLE

Write pseudocode to solve the following problem:
Given a string containing different types of parentheses {},[],() , to check if the parentheses in the input string are balanced. A string is considered balanced if:

- Every opening parenthesis has a corresponding closing parenthesis.
- Parentheses are closed in the correct order
- Example Input:
- "(){}[]" → **Legal Input**
- "{[()]}" → Legal Input
- "{[(])}" → **Illegal Input**
- "((())" → **Illegal Input**

You must use only Stack ADT methods to solve this problem.

```
function isBalanced(expression){

brackets matching_bracket = {')': '(', '}': '{', ']': '['}

if char is an opening bracket ('(', '{', '[')
{
       stack.push(char)
}
elif char is a closing bracket (')', '}', ']')
{
    if stack.isEmpty() or stack.top() != matching_bracket[char]
    {
         return False
    }
stack stack.pop()
}

return stack.isEmpty()

}
```