# Algorithm Format: Nassi-Schneiderman Diagram

A Nassi-Schneiderman Diagram (NSD) models the processing flow of a module, program, or system. It is a graphical representation of the steps that describe the processing performed by a module, program, or system.

## Symbols and Semantics

A NSD is a series of nested sequence, selection, and iteration symbols within a rectangle that describe the processing flow of one module, program or system. The nested symbol at the top of the rectangle is processed (executed) first, followed by the nested symbol immediately below the top symbol, and so on, until the symbol at the bottom of the rectangle has been processed, at which point the algorithm ends.

|  |  |
| --- | --- |
| **Symbol** | **Semantics** |
|  | A *sequence* is a rectangle that contains a verb-phrase.   * After the verb-phrase is performed, flow continues to the next nested symbol in the NSD. |
|  | A *selection* is a "back-of-an-envelope" shape used to represent a question (aka: test, condition) that usually result in a yes/no or true/false answer.   * When the answer to the question is yes/true, the nested symbols underneath the yes/true are processed. * When the answer to the question is no/false, the nested symbols underneath the no/false are processed. * Alternatively, a selection may include more than two answers; in which case only one of the answers can be true at a given time (the shape looks like a lopsided back-of-an-envelope). The nested symbols underneath the answer-that-is-true are processed. * After the nested symbols underneath the answer-that-is-true are performed, flow continues with the next nested symbol contained within the NSD. |
|  | An *iteration* is a "carpenter's square" shape used to represent a loop statement.   * There are two iteration symbols in NSD:   + A loop where the question (aka: test, condition) is tested before the loop body   + A loop where the question is tested after the loop body. * When the question results in yes/true the nested symbols inside the “carpenter’s square” (i.e., the loop body) are executed again. * When the question results in no/false the iteration ends and the nested symbol that follows the iteration symbol is then processed. |

## Nassi-Schneiderman Diagram Examples (Topics 1-4)

|  |  |
| --- | --- |
| **NSD** | **Python Code** |
| # Write a program to calculate the volume and surface area of a sphere from its radius, given as input. | |
| v2-171-07b-NSD-ch3-1.wmf | import math  radius = input("Enter radius of sphere:")  radius = eval(radius)  volume = 4 / 3 \* math.pi \* radius \*\* 3  area = 4 \* math.pi \* radius \*\* 2  print("sphere volume is", volume, \  "and area is", area) |
| # Draw a square using Turtle graphics. | |
| 04-NSD-drawSquare.wmf | import turtle  wn = turtle.Screen()  alex = turtle.Turtle()  for i in [0,1,2,3]:  alex.forward(50)  alex.left(90)  wn.exitonclick() |

## Nassi-Schneiderman Diagram Examples (Topics 5-6)

|  |  |
| --- | --- |
| **NSD** | **Python Code** |
| # Display the lyrics to "Happy Birthday", given a person’s name. | |
| v2-171-07b-NSD-ch6-happy-4.wmf | def happy():  print("Happy birthday to you!")  def sing(person):  happy()  happy()  print("Happy birthday, dear", \  person + ".")  happy()  def main():  sing("Fred")  print()  sing("Lucy")  print()  sing("Elmer") |

## Nassi-Schneiderman Diagram Examples (Topics 7-9)

|  |  |
| --- | --- |
| **NSD** | **Python Code** |
| # Display a warning message depending on the current temperature (version 1). | |
|  | # 07-tempWarning-1.py  #pre: User ready to enter a temperature.  #post: Returned the entered temperature value.  def getTemperature(prompt):  return eval(input(prompt))  #pre: degrees is a fahrenheit temperature.  #post: A message has been displayed when the  # value of degrees is within a certain  # range.  def tempWarning(degrees):  if degrees > 90:  print(  "It's really hot out there. Be careful!")  if degrees > 65 and degrees < 80:  print("Enjoy the pleasant temperature!")  if degrees < 30:  print("Brrrrr. Be sure to dress warmly!")  def main():  fahrenheit = getTemperature(  "Enter a Fahrenheit temperature: ")  tempWarning(fahrenheit) |
| # Display a warning message depending on the current temperature (version 2). | |
|  | # 07-tempWarning-2.py  #pre: User ready to enter a temperature.  #post: Returned the entered temperature value.  def getTemperature(prompt):  return eval(input(prompt))  #pre: degrees is a fahrenheit temperature.  #post: A message has been displayed when the  # value of degrees is within a certain  # range.  def tempWarning(degrees):  if degrees > 90:  print(  "It's really hot out there. Be careful!")  elif degrees > 65 and degrees < 80:  print("Enjoy the pleasant temperature!")  elif degrees < 30:  print("Brrrrr. Be sure to dress warmly!")  def main():  fahrenheit = getTemperature(  "Enter a Fahrenheit temperature: ")  tempWarning(fahrenheit) |