V. Rego

Jan 28, 2015

Week 3, Examples 2  Note: From Chapter 4 we are jumping into the first part of Chapter 7. On completing this part we will move to Chapter 5]

```python
# #1.py

# If-statement

def wait():
    x = input()
    print(" ")

def main():
    # input any non-negative number to print “Larry”
    number = eval(input("Enter any number: "))
    wait()
    if (number >= 0):
        print(" Hello Larry!")

# #2.py

# If-else-statement (2-way decision, fork in the road)

def wait():
    x = input()
    print(" ")

def main():
    # input any non-negative number to print “Larry”, else print “Moe”
    number = eval(input("Enter any number: "))
    wait()
    if (number >= 0):
        print(" Hello Larry!")
    else:
```

Computer Science Courses - http://courses.cs.purdue.edu/
print ("Hello Moe!")

main()

#_3.py ' T——- | T ———-| | | | F——- —————| | T——- | | F ———-| | F——- '# Nested If-else-statement (4-way decision, each fork leads to another fork # in the road. By road we mean “execution path taken by the CPU” based on # conditions being True or False) def wait(): x = input() print(" ") def main(): # input any number # number > 10 prints “Larry” # 0 <= number <= 10 prints “Moe” # number < -10 prints “Curly” # -10 <= number < 0 prints “Shemp” # These print statements are only used to show how the “if-else” works. # In general there will be statements and function calls, etc., in each block, # depending on what you want to have done. # NOTE: Be careful. When you test for conditions, make sure you account for # ALL the different cases, i.e., all the possible paths. Otherwise the program # may take a path that you did not expect, and it will take some effort to # trace this later — to find which condition(s) you missed. number = eval(input("Enter any number: ")) wait() if (number >= 0): if (number > 10): print(" Hello Larry!") else: print(" Hello Moe!") else: # if we are here, it means number < 0 if (number < -10): print(" Hello Curly!") else: print(" Hello Shemp!") main()__ #4.py

#Same example as in 3.py, except that we use “elif” instead of nested if-else

# Nested If-else-statement (4-way decision, each fork leads to another fork # in the road. By road we mean “execution path taken by the CPU” based on # conditions being True or False) def wait(): x = input() print(" ") def main(): # input any number # number > 10 prints “Larry” # 0 <= number <= 10 prints “Moe” # number < -10 prints “Curly” # -10 <= number < 0 prints “Shemp” # These print statements are only used to show how the “if-else” works. # In general there will be statements and functions etc in each block, # depending on what you want to have done. # NOTE: Be careful. When you test for conditions, make sure you account for # ALL the different cases, i.e., all the possible paths. Otherwise the program # may take a path that you did not expect, and it will take some effort to # trace this later — to find which condition(s) you missed. number = eval(input("Enter any number: ")) wait() if (number >10):
    print(" Hello Larry!")
elif (0 <= number <= 10):
    print(" Hello Moe!")
elif (number < -10):
    print(" Hello Curly!")
else:  # notice how the "else" at end catches all the remaining cases
    print(" Hello Shemp!")

main()
</code>

From:  
http://courses.cs.purdue.edu/ - Computer Science Courses

Permanent link:  

Last update: 2016/01/26 10:20