Khoi Duong

Prof. Yang

CE450

12/6/2022

HW#5

GitHub link: <https://github.com/MynameisKoi/CE450/tree/main/HW%235>



Source code: <https://github.com/MynameisKoi/CE450/blob/main/HW%235/1.py>

class link:

empty = ()

def \_\_init\_\_(*self*, *first*, *rest*=empty):

# isinstance(a, A): whether "a" is one of instances of A

assert *rest* is link.empty or isinstance(*rest*, link)

*self*.first = *first*

*self*.rest = *rest*

def \_\_repr\_\_(*self*):

if *self*.rest:

rest\_repr = ', ' + repr(*self*.rest)

else:

rest\_repr = ''

return 'link(' + repr(*self*.first) + rest\_repr + ')'

def \_\_str\_\_(*self*):

string = '<'

while *self*.rest is not link.empty:

string += str(*self*.first) + ', '

*self* = *self*.rest

return string + str(*self*.first) + '>'

def empty(*s*):

return *s* is link.empty

def cntn\_link(*s*, *elm*):

# Return True if elm is in the linked list s

# s: linked list

# elm: element

# return: True or False

# pass empty (null) values

if empty(*s*):

return False

# if elm is in the linked list s, return True

elif *s*.first == *elm*:

return True

# if elm is not in the linked list s, return False

else:

return cntn\_link(*s*.rest, *elm*)

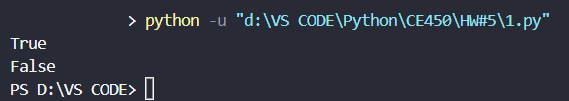
a = cntn\_link(link(1, link(2, link(3))), 1)

print(a)

b = cntn\_link(link(1, link(2, link(3))), 4)

print(b)

Run program & result:





Source code: <https://github.com/MynameisKoi/CE450/blob/main/HW%235/2.py>

class link:

empty = ()

def \_\_init\_\_(*self*, *first*, *rest*=empty):

# isinstance(a, A): whether "a" is one of instances of A

assert *rest* is link.empty or isinstance(*rest*, link)

*self*.first = *first*

*self*.rest = *rest*

def \_\_repr\_\_(*self*):

if *self*.rest:

rest\_repr = ', ' + repr(*self*.rest)

else:

rest\_repr = ''

return 'link(' + repr(*self*.first) + rest\_repr + ')'

def empty(*s*):

return *s* is link.empty

def prnt\_lnk(*s*):

string = '<'

while *s*.rest is not link.empty:

string += str(*s*.first) + ' '

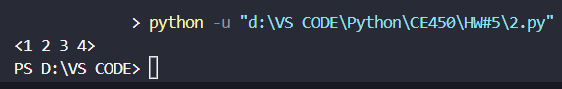
*s* = *s*.rest

return string + str(*s*.first) + '>'

a = link(1,link(2,link(3,link(4))))

print(prnt\_lnk(a))

Run program & result:





Source code: <https://github.com/MynameisKoi/CE450/blob/main/HW%235/3.py>

class link:

empty = ()

def \_\_init\_\_(*self*, *first*, *rest*=empty):

# isinstance(a, A): whether "a" is one of instances of A

assert *rest* is link.empty or isinstance(*rest*, link)

*self*.first = *first*

*self*.rest = *rest*

def \_\_repr\_\_(*self*):

if *self*.rest:

rest\_repr = ', ' + repr(*self*.rest)

elif *self*.rest is link.empty:

rest\_repr = ',[]'

else:

rest\_repr = ''

return '[' + repr(*self*.first) + rest\_repr + ']'

def empty(*s*):

return *s* is link.empty

def rvrs\_lnk(*s*):

# Return a reversed linked list

# s: linked list

# return: reversed linked list

# pass empty (null) values

if empty(*s*):

return *s*

# if s is not empty, reverse the linked list

else:

# create a new linked list

new\_lnk = link(*s*.first)

# while s is not empty, reverse the linked list

while *s*.rest is not link.empty:

# add the first element of s to the new linked list

new\_lnk = link(*s*.rest.first, new\_lnk)

# move to the next element of s

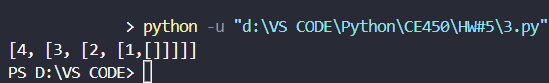
*s* = *s*.rest

return new\_lnk

a = link(1,link(2,link(3,link(4))))

print(rvrs\_lnk(a))

Run program & result:





Source code: <https://github.com/MynameisKoi/CE450/blob/main/HW%235/4.py>

class link:

empty = ()

def \_\_init\_\_(*self*, *first*, *rest*=empty):

# isinstance(a, A): whether "a" is one of instances of A

assert *rest* is link.empty or isinstance(*rest*, link)

*self*.first = *first*

*self*.rest = *rest*

def \_\_repr\_\_(*self*):

if *self*.rest:

rest\_repr = ', ' + repr(*self*.rest)

elif *self*.rest is link.empty:

rest\_repr = ',[]'

else:

rest\_repr = ''

return '[' + repr(*self*.first) + rest\_repr + ']'

def empty(*s*):

return *s* is link.empty

def srt(*link*):

# if the linked list is sorted, return true

if *link*.rest is *link*.empty:

return True

# if the linked list is not sorted, return false

elif *link*.first > *link*.rest.first:

return False

# if the linked list is not empty, return srt(link.rest)

else:

return srt(*link*.rest)

lnk1 = link(1,link(2,link(3,link(4))))

print(srt(lnk1))

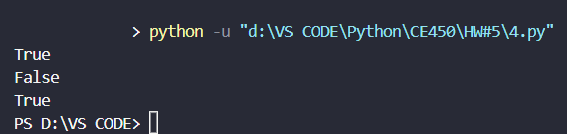
lnk2 = link(1, link(3, link(2, link(4, link(5)))))

print(srt(lnk2))

lnk3 = link(3, link(3, link(3)))

print(srt(lnk3))

Run program & result:





Source code: <https://github.com/MynameisKoi/CE450/blob/main/HW%235/5.py>

class link:

empty = ()

def \_\_init\_\_(*self*, *first*, *rest*=empty):

# isinstance(a, A): whether "a" is one of instances of A

assert *rest* is link.empty or isinstance(*rest*, link)

*self*.first = *first*

*self*.rest = *rest*

def \_\_repr\_\_(*self*):

if *self*.rest:

rest\_repr = ', ' + repr(*self*.rest)

elif *self*.rest is link.empty:

rest\_repr = ',[]'

else:

rest\_repr = ''

return '[' + repr(*self*.first) + rest\_repr + ']'

def empty(*s*):

return *s* is link.empty

def sum\_lnk(*lnk*, *g*):

# apply function g to each element in the linked list and return the sum

if empty(*lnk*):

return 0

else:

return *g*(*lnk*.first) + sum\_lnk(*lnk*.rest, *g*)

dbl = lambda *y*: *y*\*2

sqr = lambda *x*: *x*\**x*

lnk1 = link(1,link(2,link(3,link(4))))

print(sum\_lnk(lnk1, sqr))

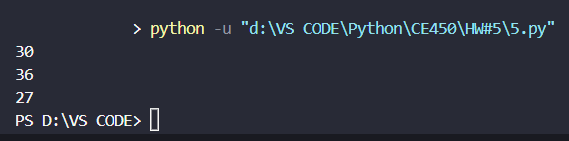
lnk2 = link(3, link(5, link(4, link(6))))

print(sum\_lnk(lnk2, dbl))

lnk3 = link(3, link(3, link(3)))

print(sum\_lnk(lnk3, sqr))

Run program & result:





Source code: <https://github.com/MynameisKoi/CE450/blob/main/HW%235/6.py>

class link:

empty = ()

def \_\_init\_\_(*self*, *first*, *rest*=empty):

# isinstance(a, A): whether "a" is one of instances of A

assert *rest* is link.empty or isinstance(*rest*, link)

*self*.first = *first*

*self*.rest = *rest*

def \_\_repr\_\_(*self*):

if *self*.rest:

rest\_repr = ', ' + repr(*self*.rest)

elif *self*.rest is link.empty:

rest\_repr = ',[]'

else:

rest\_repr = ''

return '[' + repr(*self*.first) + rest\_repr + ']'

def empty(*s*):

return *s* is link.empty

def change(*lnk*, *u*, *v*):

if empty(*lnk*):

return *lnk*

# if lnk is not empty, change the linked list

else:

if *lnk*.first == *u*:

*lnk*.first = *v*

return link(*lnk*.first, change(*lnk*.rest, *u*, *v*))

l = link(1, link(2, link(3)))

print(l)

n = change(l, 3, 1)

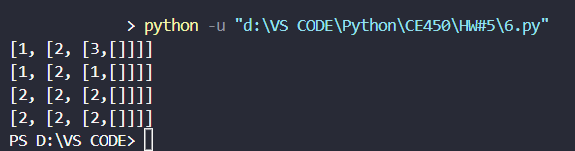
print(n)

m = change(n, 1, 2)

print(m)

print(change(m, 5, 1))

Run program & result:





Source code: <https://github.com/MynameisKoi/CE450/blob/main/HW%235/7.py>

class link:

empty = ()

def \_\_init\_\_(*self*, *first*, *rest*=empty):

# isinstance(a, A): whether "a" is one of instances of A

assert *rest* is link.empty or isinstance(*rest*, link)

*self*.first = *first*

*self*.rest = *rest*

def \_\_repr\_\_(*self*):

if *self*.rest:

rest\_repr = ', ' + repr(*self*.rest)

elif *self*.rest is link.empty:

rest\_repr = ',[]'

else:

rest\_repr = ''

return '[' + repr(*self*.first) + rest\_repr + ']'

def empty(*s*):

return *s* is link.empty

def first(*s*): # first function for class link

assert *s* is not link.empty

return *s*.first

def rest(*s*): # rest function for class link

assert *s* is not link.empty

return *s*.rest

def apnd(*lnk*, *m*):

# add the element m to the end of the linked list

# lnk: linked list

# m: element to be added

# return: linked list with m added to the end

# pass empty (null) values

if empty(*lnk*):

return link(*m*)

# if lnk is not empty, add m to the end of the linked list

else:

# create a new linked list

new\_lnk = link(*lnk*.first)

# while lnk is not empty, add m to the end of the linked list

while *lnk*.rest is not link.empty:

# add the first element of lnk to the new linked list

new\_lnk = link(*lnk*.rest.first, new\_lnk)

# move to the next element of lnk

*lnk* = *lnk*.rest

# add m to the end of the linked list

new\_lnk = link(*m*, new\_lnk)

# reverse the new\_lnk to get the final linked list

result\_lnk = link(new\_lnk.first)

while new\_lnk.rest is not link.empty:

result\_lnk = link(new\_lnk.rest.first, result\_lnk)

new\_lnk = new\_lnk.rest

return result\_lnk

l = link(1, link(2, link(3)))

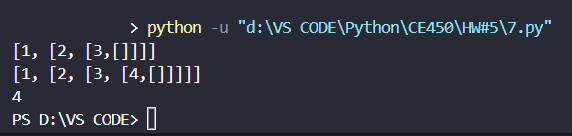
print(l)

n = apnd(l, 4)

print(n)

print(first(rest(rest(rest(n))))) # 4

Run program & result:





Source code: <https://github.com/MynameisKoi/CE450/blob/main/HW%235/8.py>

class link:

empty = ()

def \_\_init\_\_(*self*, *first*, *rest*=empty):

# isinstance(a, A): whether "a" is one of instances of A

assert *rest* is link.empty or isinstance(*rest*, link)

*self*.first = *first*

*self*.rest = *rest*

def \_\_repr\_\_(*self*):

if *self*.rest:

rest\_repr = ', ' + repr(*self*.rest)

elif *self*.rest is link.empty:

rest\_repr = ',[]'

else:

rest\_repr = ''

return '[' + repr(*self*.first) + rest\_repr + ']'

def empty(*s*):

return *s* is link.empty

def insrt(*l*, *elm*, *ind*):

# insert elm into linked list l at index ind

# l: linked list

# elm: element

# ind: index

# return: linked list

# pass empty (null) values

if empty(*l*):

return link(*elm*)

# if ind is 0, insert elm at the beginning of the linked list

elif *ind* == 0:

return link(*elm*, *l*)

# if ind is not 0, insert elm at the index ind

else:

return link(*l*.first, insrt(*l*.rest, *elm*, *ind*-1))

l = link(11, link(12, link(13)))

n = insrt(l, 2021, 1)

print(n)

m = insrt(n, 2022, 20)

print(m)

Run program & result:

