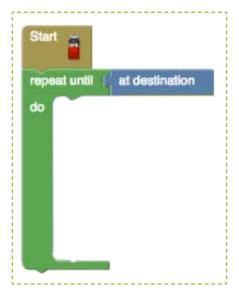
Blockly Cards

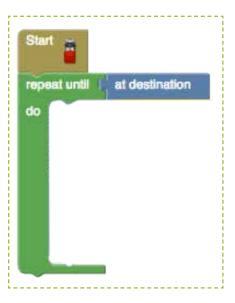


Blockly Cards

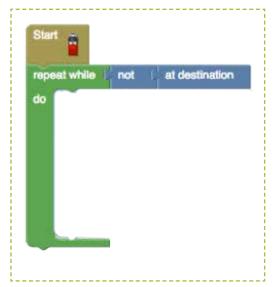
Road exists forward Road exists left Road exists right if... do... road exists left road exists forward road exists right do do road exists forward road exists left road exists right do do road exists forward road exists left road exists right else if else if do do road exists forward road exists left road exists right -* road exists left road exists forward road exists right do do else if else if road exists left do road exists forward do road exists right else if else do do road exists forward road exists left road exists right -

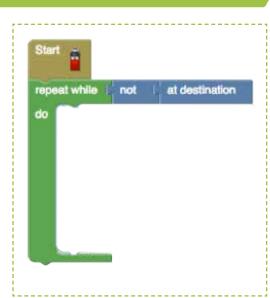
Repeat until do

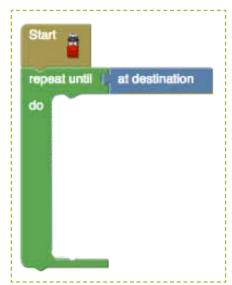


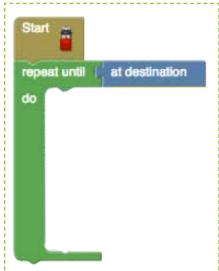


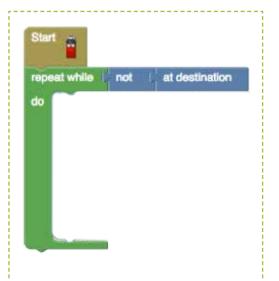
Repeat while do

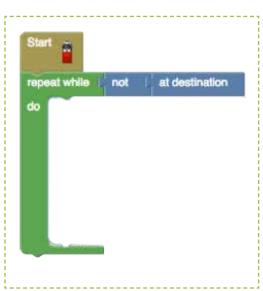












UKS2-Assets

Define do Call Define Wiggle Define Wiggle Define wiggle wiggle wiggle Do Do Do wiggle wiggle Define wiggle Define Wiggle Define wiggle wiggle Do Do Do wiggle wiggle

UKS2-Assets - Python table top code cards

Sequence instructions

v·move_forwards()

v.turn_right()

v.turn_left()

v.wait()

v·move_forwards()

v·turn_right()

v·turn_left()

v.wait()

v·move_forwards()

v.turn_right()

v.turn_left()

v.wait()

v·move_forwards()

v.turn_right()

v•turn_left()

v.wait()

Repetition

for count in range(3):

while v.at_destination():

while not v.at_destination():

for count in range(3):

while v.at_destination():

while not v.at_destination():

for count in range(3):

while v.at_destination():

while not v.at_destination():

UKS2-Assets - Python table top code cards

Selection	Procedures		Variables
if():	Define	Define	length = 10
elif():	def procname():	def procname():	length = length + 5
else:	Call	Call	
	procname()	procname()	length = 10
if():	!		length = length + 5
elif():	Define	Define	length = 10
else:	def procname():	def procname():	length = length + 5
	Call	Call	
if():	procname()	procname()	length = 10
elif():	!		length = length + 5
else:			