**The laboratory work 12**

**Wumpus world description**

In general, you can take actions that change the direction you are facing, move forward and either pick things up or shoot and arrow. Once you have scored some points and returned to your home, you can also leave the game.

Moves that change your orientation: Up, Down, Left, Right

The move that changes your position: Step

The move that allows you to exit the board: Exit

Actions that change the world aside from your position: PickUp, Shoot

You invoke these actions by calling the function take\_action (in the file updatewumpus.py) that takes the name of a world and a move. It returns a list of strings that describes the current perceptions that are available to you: (smell, air, glitter, bump, scream, location, orientation, status, score).

The possible values of these perceptions are:

**Smell** – clean|nasty

**Air** – calm|breeze

**Glitter** – bare|glitter

**Bump** – no\_bump|bump

**Scream** – quite|scream

**Location** – unique identifier for current square

**Orientation** – the direction you are facing

**Status** – living|dead|won

**Score** – current score

Before you start the game, you need to initialize a “world”. You do this by calling initialize\_world, a function that has no arguments. This returns the name of a new world that you need to use when calling take\_action. This function builds a folder (“WumpusWorldDataFolder”) where the world data will be saved. It does not delete any of the world it builds, so you might want to clear this folder occasionally.

The code for building the world of Wumpus is represented below.

This is the file – updatewumpusNowWithRocks.py

|  |
| --- |
| import random, json, os  # We are assuming the following moves:  # Moves that change your orientation  # Up  # Down  # Left  # Right  # A move that changes your position  # Step  # The move that allows you to exit the board  # Exit  # Other actions that change the world aside from your position  # PickUp  # Shoot  # The function update will return a vector of strings that represent:  # (smell, air, glitter, bump, scream, location, orientation, status, score)  # The possible values are:  # Smell - clean|nasty  # Air - calm|breeze  # Glitter - bare|glitter  # Bump - no\_bump|bump  # Scream - quiet|scream  # Location - unique identifier for current square  # Orientation - the direction you are facing  # Status - living|dead|won  # Score - current score  # You get precept vectors by calling take\_action with the name of your world and the  # move you want to take.  def take\_action(world\_token,move):  world = get\_world(world\_token)  location = world["location"]  orientation = world["orientation"]  points = world["points"]  status = world["status"]  arrows = world["arrows"]    print "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n"    if status == "dead":  print "You are dead. Start a new game"  return  elif move == "Exit":  update = update\_location(world, location, orientation)  if world["location"] != "Cell 11":  print "You need to get back to Cell 11 to exit"  elif world["points"] == 0:  print "You need to score some points in order to exit"  else:  update[7] = "won"  elif move == "Toss":  if world["rocks"] <= 0:  print "You are out of rocks"  return  else:  world["rocks"] = world["rocks"] - 1  print "Tossing a rock. You have " + str(world["rocks"]) + " left."  store\_world(world\_token,world)  cell\_state = world[world[location][orientation]]  if cell\_state["Pit"] is True:  return "Quiet"  else:  return "Clink"  elif move == "Step":  print "Taking a step"  new\_location = world[location][orientation]  if new\_location == "Void":  print "You bumped your head on the edge of the world."  update = update\_location(world, location, orientation)  update[3] = "bump"  else:  print "Moving to " + str(new\_location)  update = update\_location(world, new\_location, orientation)  world["location"] = new\_location  elif move in["Up","Down","Left","Right"]:  print "Turing to face " + move  update = update\_location(world, location, move)  world["orientation"] = move  elif move == "PickUp":  print "Trying to pick up gold"  if got\_gold(world, location):  print "You've picked up some gold!"  print "You get 1000 more points!"  update = update\_location(world, location, orientation)  update[2] = "bare"  world[location]["Gold"] = False  world["points"] = world["points"]+1000  else:  print "There is no gold here!"  update = update\_location(world, location, orientation)  elif move == "Shoot":  print "Trying to shoot the Wumpus"  if world["arrows"] <= 0:  print "You are out of arrows"  elif wumpus\_in\_sight(world,location, orientation):  print "You killed the Wumpus!"  print "You get 100 more points!"  wumpus\_location = where\_is\_the\_Wumpus(world,location,orientation)  world[wumpus\_location]["Wumpus"] = False  world["points"] = world["points"]+100  else:  print "You missed the Wumpus!"  update = update\_location(world, location, orientation)  world["arrows"] = world["arrows"] - 1  print "Perception = (" + ", ".join(update) + ")"  world["status"] = update[7]  store\_world(world\_token,world)  update[8] = world["points"]  return update  # Update\_location figures out the perceptual elements associated with a location by  # checking for gold, pits and the Wumpus  def update\_location(world,location,orientation):  location\_info = world[location]  baseline = ["clean","calm","bare","no\_bump","quiet",location, orientation, "living", str(world["points"])]    if got\_gold(world, location):  print "There is a lovely glitter in the room"  baseline[2] = "glitter"    if got\_breeze(world, location):  print "There is a breeze running through this room"  baseline[1] = "breeze"    if got\_smell(world, location):  print "There is a nasty smell in here"  baseline[0] = "nasty"    if location\_info["Wumpus"] is True:  print "You got killed by the Wumpus and it was shockingly painful"  baseline[7] = "dead"    elif location\_info["Pit"] is True:  print "You fell into a pit and died a slow and scary death"  baseline[7] = "dead"  return baseline  # Various tests to figure out precept list.  # Is there gold in this cell?  def got\_gold(world, location):  return world[location]["Gold"]  # Do any of the adjacent cells have Pits in them?    def got\_breeze(world,location):  for x in world[location]["Next"]:  if world[x]["Pit"]:  return True  return False  # Do any of the adjacent cells have the Wumpus?    def got\_smell(world,location):  for x in world[location]["Next"]:  if world[x]["Wumpus"] is True:  return True  return False      # Is there are Wumpus in the agent's line of sight?    def wumpus\_in\_sight(world, location, orientation):  next\_location = world[location][orientation]  if next\_location == "Void":  return False  elif world[location]["Wumpus"] is True:  return True  else:  return wumpus\_in\_sight(world, next\_location, orientation)  # Where is the Wumpus in the agent's line of sight?    def where\_is\_the\_Wumpus(world, location, orientation):  next\_location = world[location][orientation]  if world[location]["Wumpus"] is True:  return location  else:  return where\_is\_the\_Wumpus(world, next\_location, orientation)  # look\_ahead  def look\_ahead(world\_token):  world = get\_world(world\_token)  return world[world["location"]]["Next"]  # Build out the dictionary that makes up the simple world that we have been looking at    def build\_world(gold, wumpus, pits):  layout = {}  height = 4  width = 4  for x in range(1,width+1):  for y in range(1,height+1):  new\_cell = {}  new\_cell["Up"] = "Void"  new\_cell["Down"] = "Void"  new\_cell["Left"] = "Void"  new\_cell["Right"] = "Void"  new\_cell["Wumpus"] = False  new\_cell["Pit"] = False  new\_cell["Gold"] = False  new\_cell["Next"] = []  if y < 4:  new\_cell["Up"] = "Cell " + str(x) + str(y+1)  new\_cell["Next"].append(new\_cell["Up"])  if y > 1:  new\_cell["Down"] = "Cell " + str(x) + str(y-1)  new\_cell["Next"].append(new\_cell["Down"])  if x < 4:  new\_cell["Right"] = "Cell " + str(x+1) + str(y)  new\_cell["Next"].append(new\_cell["Right"])  if x > 1:  new\_cell["Left"] = "Cell " + str(x-1) + str(y)  new\_cell["Next"].append(new\_cell["Left"])  layout["Cell "+str(x)+str(y)] = new\_cell  layout[wumpus]["Wumpus"]=True  print "There is a Wumpus in cell " + wumpus + "."  layout[gold]["Gold"]=True  print "There is Gold in cell " + gold + "."  for cell in pits:  layout[cell]["Pit"]=True  print "There is a Pit in cell " + cell + "."  print  return layout  # In order to have a persistant world, we are going to store and update it as the  # game progresses.  # We first initialize the state of the world and return a random token to the user  # so that they can refer to the world that they are playing in  def intialize\_world():  world\_name = "Wumpus" + str(random.randint(0,10000))  print "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n"  print "Initializing your new Wumpus world!"  print "Your new world is called: "+ world\_name  if not os.path.exists("WumpusWorldDataFolder"):  os.makedirs("WumpusWorldDataFolder")  world=build\_world("Cell 32", "Cell 13", ["Cell 31","Cell 33","Cell 44"])  world["location"] = "Cell 11"  world["orientation"] = "Right"  world["status"] = "living"  world["points"] = 0  world["arrows"] = 1  world["rocks"] = 5  print "You are starting in Cell 11, looking to the Right."  print "You are starting with 0 points, " + str(world["arrows"]) + " arrow(s)."  print "You have " + str(world["rocks"]) + " rocks."  print "You are alive."  with open("WumpusWorldDataFolder/"+world\_name+".json", 'w') as worldfile:  json.dump(world, worldfile)  worldfile.close()  return world\_name  # In order to have a persistant world, we are going to store and update it as the  # game progresses.  # We first initialize the state of the world and return a random token to the user  # so that they can refer to the world that they are playing in  def intialize\_my\_world(gold,wumpus,pits):  world\_name = "Wumpus" + str(random.randint(0,10000))  print "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n"  print "Initializing your own Wumpus world!"  print "Your new world is called: "+ world\_name  if not os.path.exists("WumpusWorldDataFolder"):  os.makedirs("WumpusWorldDataFolder")  world=build\_world(gold,wumpus,pits)  world["location"] = "Cell 11"  world["orientation"] = "Right"  world["status"] = "living"  world["points"] = 0  world["arrows"] = 1  world["rocks"] = 5  print "You are starting in Cell 11, looking to the Right."  print "You are starting with 0 points, " + str(world["arrows"]) + " arrow(s)."  print "You have " + str(world["rocks"]) + " rocks."  print "You are alive."  with open("WumpusWorldDataFolder/"+world\_name+".json", 'w') as worldfile:  json.dump(world, worldfile)  worldfile.close()  return world\_name  # At the beginning of each turn, we load the last state of the world so we know  # what precepts to return in response the actions.    def get\_world(world\_name):  with open("WumpusWorldDataFolder/"+world\_name+".json") as worldfile:  world = json.load(worldfile)  worldfile.close()  return world  # As things change in response to actions, we update and store the world in response to  # actions that have been taken  def store\_world(world\_name,world):  with open("WumpusWorldDataFolder/"+world\_name+".json", 'w') as worldfile:  json.dump(world, worldfile)  worldfile.close() |