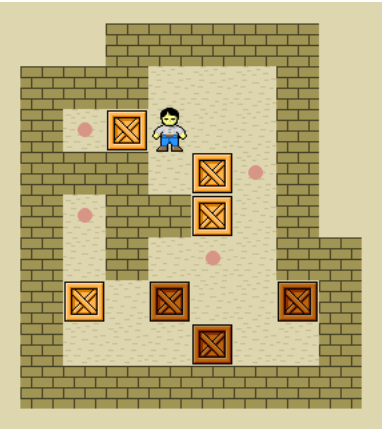
Problem 1. *Sokoban là gì ?*

Answer.

- Sokoban là trò chơi dạng câu đố trong đó người chơi phải đẩy một số khối vuông vượt qua chướng ngại vật để đến đích. Trò chơi đã được thiết kế vào năm 1981 bởi Hiroyuki Imabayashi và được ra mắt lần đầu vào tháng 12 năm 1982.



Hình 1: Sokoban Game from Wikipedia

- Trong bài tập này, ta sẽ lần lượt sử dụng các thuật toán Uninformed Search như DFS, BFS và UCS để giải và so sánh độ hiệu quả của từng thuật toán trên 18 màn chơi.

Problem 2. *Sokoban đã được mô hình hóa như thế nào ?*

1. Biểu diễn bài toán

- Các màn chơi (level) sẽ được lưu dưới dạng các txt files, sau đó sẽ được load lên để hiện thị dưới dạng đồ họa 2D bằng cách convert những kí tự trong file text thành một ma trận 2 chiều các số nguyên dương tương ứng.

```
######
#.  .#
#    #
# BB #
#&   #
######
```

```
######
#.  .#
#    #
#B  B#
#&   #
######
```

- Trong đó các kí hiệu tương ứng:
  - "#" là bức tường được biểu diễn là 1.

– "B" là các box chưa nằm đúng vị trí được biểu diễn là 3.

– "X" là các box đã nằm đúng goal được biểu diễn là 5.

– "." chính là các goal mà ta cần phải đẩy box vào được biểu diễn là 4.

– "&" chính là vị trí bắt đầu của người chơi được biểu diễn là 2.

– " " là các vùng trống được biểu diễn là 0

Code dùng để convert:

```
for irow in range(len(layout)):
  for icol in range(len(layout[irow])):
    if layout[irow][icol] == ' ': layout[irow][icol] = 0
    elif layout[irow][icol] == '#': layout[irow][icol] = 1
    elif layout[irow][icol] == '&': layout[irow][icol] = 2
    elif layout[irow][icol] == 'B': layout[irow][icol] = 3
    elif layout[irow][icol] == '.': layout[irow][icol] = 4
    elif layout[irow][icol] == 'X': layout[irow][icol] = 5
colsNum = len(layout[irow])
```

2. Các yếu tố của bài toán tìm kiếm trong Sokoban

- Mỗi state mang thông tin liên quan đến vị trí hiện tại của người chơi và boxes
- Initial State là vị trí bắt đầu của player và boxes được cung cấp
- Trạng thái kết thúc là thời điểm tất cả các goal đều được filled bởi các boxes
- Actions: người chơi có thể di chuyển theo 4 hướng (Up, Down, Left, Right) ngoài ra còn có hành động đẩy các boxes (1 ô).
- Từ một state hiện tại, ta có thể sinh ra tối đa 4 state con tùy thuộc vào việc các hành động được tạo ra đó có hợp lệ hay không. Ta có thể xây dựng một kiến trúc cây với các node là những state của bài toán và node gốc là initial state.
- Các hành động không hợp lệ có thể bao gồm: di chuyển đụng tường, đẩy nhiều hợp chồng nhau,... Ta có thể kiểm tra tính hợp lệ của hành động thông qua hàm isLegalAction()

Ta sinh ra các hành động bằng hàm LegalActions()

```
def legalActions(posPlayer, posBox):
  """Return all legal actions for the agent in the current game
   state"""
  allActions = [[-1,0,'u','U'],[1,0,'d','D'],[0,-1,'l','L'],[0,1,
   'r','R']]
  xPlayer, yPlayer = posPlayer
  legalActions = []
  for action in allActions:
    x1, y1 = xPlayer + action[0], yPlayer + action[1]
    if (x1, y1) in posBox: # the move was a push
      action.pop(2) # drop the little letter
    else:
      action.pop(3) # drop the upper letter
    if isLegalAction(action, posPlayer, posBox):
      legalActions.append(action)
    else:
      continue

  return tuple(tuple(x) for x in legalActions) # e.g. ((0, -1, 'l
   '), (0, 1, 'R'))
```

- Kết quả trả về là 1 tuple với 3 thành phần: 2 thành phần đầu là cách di chuyển ứng với tọa độ x và y của player và một kí tự để xem bước đi đó có phải là đẩy box hay không (In hoa biểu thị cho hành động đẩy box)

Successor function: cập nhật vị trí của player và boxes sau mỗi action

```
def updateState(posPlayer, posBox, action):
  """Return updated game state after an action is taken"""
  xPlayer, yPlayer = posPlayer # the previous position of player
  newPosPlayer = [xPlayer + action[0], yPlayer + action[1]] # the
    current position of player
  posBox = [list(x) for x in posBox]
  if action[-1].isupper(): # if pushing, update the position of
   box
    posBox.remove(newPosPlayer)
    posBox.append([xPlayer + 2 * action[0], yPlayer + 2 * action
   [1]])
  posBox = tuple(tuple(x) for x in posBox)
  newPosPlayer = tuple(newPosPlayer)
  return newPosPlayer, posBox
```