# Worksheet 4 Stacks

**Task 1 Crushing cars**

1. (a) Complete the following to show the operations implemented on a collection of burnt-out cars. The stack can hold a maximum of 6 items.

**Cars:** Mondeo, Golf, Fiesta, Punto, Civic, Porsche

Representations of the stack drawn both horizontally and vertically are shown. Show the state of the stack after each push and pop operation in both representations, and in the first table, show any results returned.

|  |  |  |
| --- | --- | --- |
|  | **Stack** | **Result returned** |
| carStack = Stack() | [] |  |
| carStack.push(Mondeo) | [Mondeo] |  |
| carStack.push(Golf) | [Mondeo, Golf] |  |
| carStack.isEmpty() |  | False |
| carStack.push(Fiesta) | [Mondeo, Golf, Fiesta] |  |
| carStack.push(Punto) | [Mondeo, Golf, Fiesta, Punto] |  |
| carStack.pop() | [Mondeo, Golf, Fiesta] | Punto |
| carStack.push(Civic) | [Mondeo, Golf, Fiesta, Civic] |  |
| carStack.push(Porsche) | [Mondeo, Golf, Fiesta, Civic, Porsche] |  |
| carStack.isFull() |  | False |
| carStack.pop() | [Mondeo, Golf, Fiesta, Civic] | Porsche |
| carStack.pop() | [Mondeo, Golf, Fiesta] | Civic |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | Punto |
|  |  |  |  |  |  | Fiesta |  | Fiesta |
|  |  |  |  | Golf |  | Golf |  | Golf |
|  |  | Mondeo |  | Mondoe |  | Mondeo |  | Mondeo |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  | Porsche |  |  |  |  |
|  |  | Civic |  | Civic |  | Civic |  |  |
| Fiesta |  | Fiesta |  | Fiesta |  | Fiesta |  | Fiesta |
| Golf |  | Golf |  | Golf |  | Golf |  | Golf |
| Mondeo |  | Mondeo |  | Mondeo |  | Mondeo |  | Mondeo |

2. Complete the pseudocode below for a program which uses a stack to test an input string to determine whether it is a palindrome (the same backwards and forwards, like “peep”)

Assume that a class **Stack** implements the operations in the table in question 1.

myString = input (“Please enter a word or phrase to be tested: ”)

list1 = list(myString) //convert myString to a list of characters

numChars = len(list1)

s = Stack()

for i = 0 to numChars-1 do

s.push(list[i])

next i

endfor

list2 = list()

while s.isEmpty() == False do

list2.append(s.pop()) //puts everything into reverse order so we can check if the numbers are equal (by comparing forward and reverse order)

endwhile

if list1 == list2 then

palindrome = True

3. Implement your algorithm in Python, using a dynamic list to represent the stack. Screenshot your code below:

number = input("enter a number")

palindrome = False

list1 = []

list2 = []

for i in range(0, len(number)):

    list1.append(number[i])

for i in range(len(list1)-1, -1, -1):

    list2.append(list1[i])

for i in range(0, len(list1)):

    if list1[i] != list2[i]:

        palindrome = False

        break

    elif list1[i] == list2[i]:

        palindrome = True

if palindrome == True:

    print(number, "is a palindrome")

else:

    print(number, "is not a palindrome")