

CSEN 102

Introduction to Computer Science

Lecture 5:

Algorithmic Problem Solving Iterative Operations Over Lists

Prof. Dr. Slim Abdennadher
Dr. Mohammed Salem, slim.abdennadher@guc.edu.eg,
mohammed.salem@guc.edu.eg
German University Cairo, Department of Media Engineering and Technology

27.10.2018 - 01.11 2018

1 Synopsis

1.1 Sequential, conditional, iterative

What you should have learned so far...

Algorithms can be constructed by the following operations:

- *Sequential Operation*
- *Conditional Operation*
- *Iterative Operation*

Syntax

Conditional control flow: general format

```
1 if condition:  
2     # <operations for the then-part>  
3 else  
4     # <operations for the else-part>
```

Iterative control flow: general format

```
1 while <condition>:  
2     step 1: <operation>  
3     ...  
4     step i: <operation>
```

Syntax

By the way...

what is the control-flow syntax for *sequential* operations?

1.2 Sequence, conditional, and iteration combined

Sequence, conditional, and iteration in one algorithm

- Remember the *Euclidean Algorithm* from lecture 1, slide 24 to determine the greatest common divisor (GCD) of two integers.
- *Method*: To find the GCD of two numbers, repeatedly replace the larger by subtracting the smaller from it until the two numbers are equal.

Consider this little warm-up...

```
1 A, B = eval(input()), eval(input())
2 while not A == B:
3     if A > B:
4         A = A - B
5     else:
6         B = B - A
7 print("The_GCD_is_")
8 print(A)
```

2 Lists

2.1 Idea and notations

Lists

- A list is a *collection of data*.
- In Python, we denote a list with `[]`
- `A[i]` corresponds to the value of the item in position `i`
- To get a list of `n` elements: `A = eval(input())`
- To get its length `n`: `n = len(A)`

2.2 Introductory example

Example I

Example 1. Given a list of n numbers, where n is odd, find the middle number in the list.

```
1 list_A = eval(input())
2 n = len(list_A)
3 i = int(n/2)
4 mid = list_A[i]
5 print(mid)
```

Example II

Example 2. Given a list of numbers, find the sum of the numbers in the list.

```
1 list_A = eval(input())
2 n = len(list_A)
3 i = 0
4 result = 0
5 while i < n:
6     result = result + list_A[i]
7     i = i + 1
8
9 print(result)
```

Example III

Example 3. Given a list of numbers, find the number of times a given number occurs in the list.

```
1 number = eval(input())
2 list_A = eval(input())
3 n = len(list_A)
4 count = 0
5 i = 0
6 while (i < n):
7     if (list_A[i] == number):
8         count += 1
9     i += 1
10 print(count)
```

Example IV

Sequential search

Problem: Find the phone number of a given Name in an (unsorted) list of names and their phone numbers

Names	Phone numbers
N0	T0
N1	T1
...	...
N999	T999

Example IV

Sequential search

Problem: Find the phone number of a given Name in an (unsorted) list of names and their phone numbers

Sequential search, 1st attempt

```
1 Name = input()
2 list_N = eval(input())
3 list_T = eval(input())
4 if Name == list_N[0]: print(list_T[0])
5 if Name == list_N[1]: print(list_T[1])
6 # ...
7 if Name == list_N[999]: print(list_T[999])
```

Example IV

Sequential search, using a loop (2nd attempt)

```
1 Name = input()
2 list_N = eval(input())
3 list_T = eval(input())
4 i = 0
5 Found = False
6
7 while Found == False and i < 1000:
8     if Name == list_N[i]:
9         print(list_T[i])
10        Found = True
11    else:
12        i = i + 1
13 if Found == False:
14    print("Sorry, name is not in directory")
```

Example V

Lists: Find The Largest Number

Problem: Given a list of values $A_0, \dots, A_{(n-1)}$, find the largest value and its (first) location

- Example:

			↓				
	A0	A1	A2	A3	A4	A5	A6
Value	5	2	8	4	8	6	4

The largest number is 8 at location 2.

- Idea: Go through the entire list, at each iteration find the largest-so-far and record its location

Example V

Lists: Find The Largest Number

Problem: Given a list of values $A_0, \dots, A_{(n-1)}$, find the largest value and its (first) location

- Example:

			i				
			↓				
	A0	A1	A2	A3	A4	A5	A6
Value	5	2	8	4	8	6	4

Largest value: $A_0, 5$

Position: 0

1. Set the *largest-so-far* to the value of A_0
2. Set location to 0
3. Set i to 1

Example V

Lists: Find The Largest Number

Problem: Given a list of values $A_0, \dots, A_{(n-1)}$, find the largest value and its (first) location

- Example:

		i	i				
		↓	↓				
	A0	A1	A2	A3	A4	A5	A6
Value	5	2	8	4	8	6	4

Largest value: $A_0, 5$

Position: 0

1. Compare the entry at position i (A_1) with the current maximum
2. Since A_0 is bigger, do not update the current maximum
3. Set i to $i + 1$ (now 2)

Example V

Lists: Find The Largest Number

Problem: Given a list of values $A_0, \dots, A_{(n-1)}$, find the largest value and its (first) location

- Example:

			i	i			
			↓	↓			
	A0	A1	A2	A3	A4	A5	A6
Value	5	2	8	4	8	6	4

Largest value: $A_0, 5, A_2, 8$

Position: 0,2

1. Compare the entry at position i (A_2) with the current maximum
2. Since A_2 is bigger, *update* the current maximum
3. Set i to $i + 1$ (now 3)

Example V

Lists: Find The Largest Number

Problem: Given a list of values $A_0, \dots, A_{(n-1)}$, find the largest value and its (first) location

- Example:

				i	i	i	i
			\downarrow	\downarrow	\downarrow	\downarrow	
	A0	A1	A2	A3	A4	A5	A6
Value	5	2	8	4	8	6	4

Largest value: $A_2, 8$

Position: 2

1. Compare the entry at position i (A_3) with the current maximum
2. Since A_2 is bigger, do not update the current maximum
3. Set i to $i + 1$ (now 4)... and so on (now 5)... and so on (now 6)...

Example V

Lists: Find The Largest Number

Problem: Given a list of values $A_0, \dots, A_{(n-1)}$, find the largest value and its (first) location

- Example:

							i	i
							\downarrow	\downarrow
	A0	A1	A2	A3	A4	A5	A6	
Value	5	2	8	4	8	6	4	

Largest value: $A_2, 8$

Position: 2

1. As soon as i is larger than the number of elements in the list
2. Stop the algorithm
3. Output the *current maximum* and *position*

Example V

Largest number, python

```
1 list_A = eval(input())
2 n = len(list_A)
3 largest_so_far = list_A[0]
4 location = 0
5 i = 1
6 while i < n:
7     if list_A[i] > largest_so_far:
8         largest_so_far = list_A[i]
9         location = i
10    i = i + 1
11 print(largest_so_far)
12 print(location)
```

Example VI

List reversal I

Reverse a list in another list

```
1 list_A = eval(input())
2 n = len(list_A)
3 list_B = []
4 i = 0
5 while i < n:
6     list_B = list_B + list_A[n-(i+1)]
7     i = i + 1
8
9 print(list_B)
```

Example VI

List reversal II

Reverse a list in place

- Idea: Swap the *first* element with the *last* element and the *second* element with the *last but one* element and so on

```
1 list_A = eval(input())
2 n = len(list_A)
3 i = 0
4 while i < n//2:
5     tmp = list_A[i]
6     list_A[i] = list_A[n-(i+1)]
7     list_A[n-(i+1)] = tmp
8     i += 1
9
10 print(list_A)
```

Iteration over Strings: Example VII

Characters in Strings:

Write an algorithm to print the characters in a String one by one

```
1 word = input()
2 n = len(word) # len() gets the length of the
3 i = 0 # the first char is at position
4 while i < n:
5     print(word[i])
6     i += 1
```

Iteration over Strings: Example VIII

Reverse a String:

Write an algorithm to reverse a given String

```
1 word = input()
2 n = len(word) # len() gets the length of the String
3 i = n - 1 # the first char is at position 0
4 result = ""
5 while i >= 0:
6     result += word[i]
7     i -= 1
8 print(result)
```