CMPT 120

Topic: Commonly used algorithms using loops
Last Lecture

- Iterative Statements
  - while loop
Learning outcomes

At the end of this course, a student is expected to:

• Create (design), analyze, and explain the behaviour of simple algorithms:
  • Solve problems by designing simple algorithms, e.g., basic calculations, searching in strings and lists, counting, calculating running sums and products
Today’s Menu

• Let’s have a look at some algorithms commonly used in CS:
  • Counting: “running a count”
  • Running a sum
  • Running a product
  • Computing an average
  • Building a sequence -> a string or a list
  • Looking for
    • the maximum element in a sequence
    • the minimum element in a sequence
    • how often does a particular element occurs in a sequence
    • etc...
Loops are great when ...

- We need to accumulate a count, a sum, a product, etc. over a sequence
  - also known as running a count (or simply counting), running a sum, running a product, etc.

- Here is running a sum:
  ```python
  sum = 0
  for grade in [2.5, 4.0, 5.5, 6.0]:
    sum += grade
  print("sum is \%f" % sum)
  ```
Computing an average

Problem Statement

• Create a program that compute the average of midterms.
Loops are great when …

- We need to incrementally create a sequence such as a string or a list
- Here, we build a string:

```python
builtString = ""
for element in range(65, 68, 1):
    builtString += chr(element)
```

- However, the following Python fragment code is not very efficient
- Why?
- Answer:
Strings are immutable

```
builtString = ""
for element in range(65, 68, 1) :
    builtString += chr(element)
```
Building strings efficiently

```python
builtList = []
for element in range(65, 68, 1):
    builtList.append(chr(element))
''.join(builtList)

• Another way is to use a list comprehension
  "".join([chr(element)
              for element in range(65, 68, 1)])
```
Loops are great when …

- We are looking for a particular element in a sequence:

```python
print("Looking for largest element:")
elements = [45, 67, 23, 88, 0]
largest = elements[0]
for element in elements :
    if element > largest :
        largest = element

print("The largest element from elements %s is %d." % (elements,largest))
```
Summary

- We had a look at some algorithms commonly used in CS
Next Lecture

- Function