Problem6: BinarySearch

Akira MATSUI

March 27, 2017

Introduction

In this exercise, we will learn binary search algorithm, which is to find a specific number in a sorted list. To use this search algorithm, you have to make sure that your list is ordered numerically. This exercise is related to bubble sort exercise.

If you have a number A that you need to find out where it is in a list, binary search algorithm tell you that in following steps. Suppose the list has N elements.

- compare A with the mean number ,R_m, of the list ($m := N/2$)
- if $A > R_m$, compare A with R_{m_2} ($m_2 := 3N/4$)
- if $A < R_m$, compare A with R_{m_2} ($m_2 := N/4$)

Here is the simple explanation.

There is a list that has numbers. Suppose that you want to find where $T = 9$ is, but cannot find it at a glance. This might happens if you list has a hundreds of elements.

`(1,2,3,6,7,8,9)`

First Step

Compare T with 6. In this case, $T > 6$ then go to next step.

Divide the list into half,

`(1,2,3)`, `(7,8,9)`.

Because $Target > 6$, we are interested in the latter list `(7,8,9)`.

Second Step

You have

`(7,8,9)`.

Compare $Target$ with 8. In this case, $Target > 8$ then we have found where $Target = 9$ is in the list.

Question

- you have `ArrayToSearch <- sort(as.integer( runif(100, min = 1, max = 99) ))`
- Implement `binary search` find where `T` is in `ArrayToSearch`
- In this excersise, set `T <- as.integer( runif(1, min = 1, max = 99)`.
- In the case `ArrayToSearch` dose not have the same number as `T`, print('can't find')

Sample Answer

```
binary <- function(ArrayToSearch,Target){

  N <- length(ArrayToSearch) #Number of factor
  Start <- 1
  End   <- N

  while(Start<End){
    Mid <- as.integer((Start + End)/2)
    if(ArrayToSearch[Mid] == Target){
      return(Mid)
    }else if(Target < ArrayToSearch[Mid]){
      End <- Mid
    }else{
      Start <- Mid + 1
    }
  }
  print('Not found')
 }

ArrayToSearch <- sort(as.integer( runif(100, min = 1, max = 99) ))
Target <-  as.integer( runif(1, min = 1, max = 99) )

print(Target)
```

```
## [1] 54
```

```
ArrayToSearch[binary(ArrayToSearch,Target)]
```

```
## [1] 54
```

```
print(ArrayToSearch)
```

```
##   [1]  1  2  2  3  6  9 11 12 13 16 17 20 22 23 24 25 25 25 26 26 28 28 28
##  [24] 30 31 33 33 36 36 37 38 40 41 41 42 42 42 42 43 44 45 46 46 46 47 47
##  [47] 50 51 51 51 52 52 54 54 55 55 56 57 57 58 58 58 59 61 62 64 65 66 67
##  [70] 67 68 70 70 70 71 72 76 77 78 81 82 82 83 84 84 85 86 88 89 90 90 92
##  [93] 93 93 93 94 94 95 95 96
```