Algorithm Design

Fundamentals of Computer and Programming

Fall 2019

Bahador Bakhshi

CE & IT Department, Amirkabir University of Technology

What We Will Learn

- Sample algorithms to practice problem solving steps
- Input & Output analysis
- Algorithm design
  - Pseudo-code

محاسبه میانگین ۳ عدد

Algorithm: Average

1- print “Please enter three integers”

2- read x1, x2, x3

3- sum ← x1 + x2 + x3

4- average ← sum / 3

5- print “Average = ” average

الگوریتم تشخیص زوج یا فرد بودن عدد

Algorithm: Odd-Even-1

1- print “Please enter an integer”

2- read n

3- y ← n mod 2

4- if (y == 0)

print “Number is even”

else

print “Number is odd”

الگوریتم تشخیص زوج یا فرد بودن عدد

Algorithm: Odd-Even-2

1- print “Please enter an integer”
2- read n
3- if(n < 0)
        n ← -1 * n
4- while(n >= 2)
        n ← n – 2
5- if(n = 0)
        print “even”
   else
        print “odd”

Verify the Algorithm

الگوریتم تشخیص زوج یا فرد بودن عدد

Algorithm: Odd-Even-3

1- print “Please enter an integer”

2- read n

3- while (n >= 2) or (n <= -1)

    n ← n - sign(n) * 2

4- if (n = 1)

    print “odd”

  else

    print “even”

الگوريتمي كه يك رشته عدد را كه با 0 تمام مي‌شود را مي‌گيرد و تعداد اعداد زوج و فرد را چاپ مي‌كند

Algorithm: Count Odd-Even

```
odd_cnt ← 0
even_cnt ← 0
print “Please enter an integer”
read n
while (n != 0)
        y ← n mod 2
        if (y == 0)
                even_cnt ← even_cnt + 1
        else
                odd_cnt ← odd_cnt + 1
        print “Please enter an integer”
        read n

print “Odd = “ odd_cnt “Even = “ even_cnt
```

الگوريتمي كه يك عدد صحيح مثبت را بگيرد و مجموع ارقام آن را چاپ كند

Algorithm: Digit-Sum

```
print “Please enter a positive integer”
read n
sum ← 0
m ← n
while (n != 0)
        y ← n mod 10
        sum ← sum + y
        n ← n - y
        n ← n / 10
print “sum of digits of” m “ = “ sum
```

Verify the Algorithm

الگوريتمي كه يك عدد صحيح مثبت را بگيرد و آنرا در مبناي 8 چاپ كند

Algorithm: Base-8

```
print “Please enter a positive integer”
read n
i ← 0
while (n != 0)
        x[i] ← n mod 8
        n ← floor (n / 8)
        i ← i + 1
i ← i - 1
while (i >= 0)
        print x[i]
        i ← i - 1
```

الگوريتمي كه يك عدد صحيح مثبت را بگيرد و فاكتوريل آنرا توليد كند

Algorithm: Factorial-1

```
print “Please enter a positive integer”
read n
i ← 1
result ← 1
while (i <= n)
        result ← i * result
        i ← i + 1

return  result
```

الگوريتمي كه يك عدد صحيح مثبت را بگيرد و فاكتوريل آنرا توليد كند

Algorithm: Factorial-2

```
print “Please enter a positive integer”
read n
result ← 1
while (n > 0)
        result ← result * n
        n ← n - 1

return  result
```

الگوريتمي كه يك عدد صحيح مثبت را بگيرد و فاكتوريل آنرا توليد كند

Algorithm: Factorial-Recursive (**n**)

```
if (n == 1)
    return 1
else
    return n * Factorial-Recursive (n - 1)
```

الگوريتمي كه يك رشته عدد را كه محل عضو اول آن با start و محل عضو آخر آن با end مشخص شده است را به صورت صعودي مرتب كند.

Algorithm: sort (**x, start, end**)

```
while (start != end)
        j ← find index of minimum element from start to end
        swap x[j] and x[start]
        start ← start + 1
```

==========

Algorithm find_min(**x, start, end**)

```
i ← start
y ← i
while (i <= end)
        if(x[i] < x[y])
                y ← i
        i ← i + 1
return y
```

Verify the Algorithm

الگوريتمي كه يك رشته عدد را كه محل عضو اول آن با start و محل عضو آخر آن با end مشخص شده است را به صورت صعودي مرتب كند.

Algorithm swap(**x, j, i**)

temp ← x[j]
x[j] ← x[i]
x[i] ← temp

الگوریتمي که آرایه صعودی از اعداد صحیح را بگیرد و آنرا تبدیل به آرایه نزولی کند.

Algorithm reverse(**A, start, end**)

```
if (start >= end)
    return
else
    swap(A, start, end)
    reverse(A, start + 1, end - 1)
```

Summary

- There are more than one algorithm for a problem
  - Efficiency, Complexity, Clarity, …
- Algorithm (Programming Language) building blocks
  - Input / Output (Lecture 5)
  - Calculations (Lecture 4)
  - Decision Making (Lecture 6)
  - Repeating (Lecture 7)
  - Modular Programming (Lecture 8)
  - Arrays + Memory Management (Lectures 9 + 10)
  - Others (Files, …) (Lecture 11 + 12)