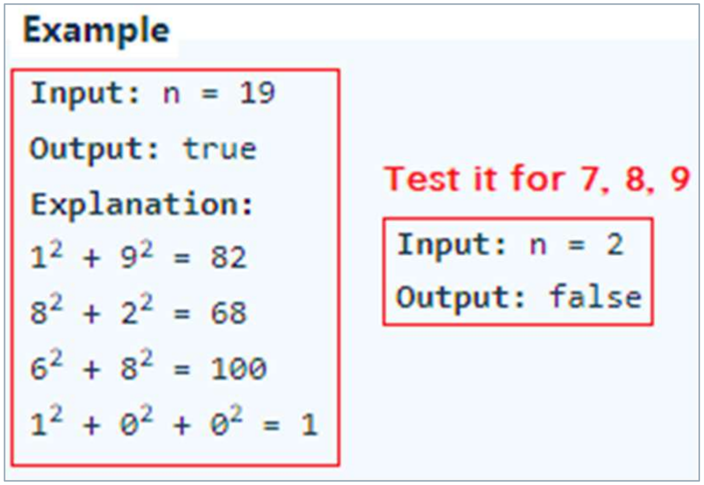
Lecture 2

Fundamentals of Algorithmic Problem Solving: The STAIR Steps for Solving Problems, and Major Factors in Designing an Algorithm.

Assignment No. 1 CLO-1

- Write an algorithm to determine if a number n is happy.
  - A happy number is a number defined by the following process:
    - Starting with any positive integer, replace the number by the sum of the squares of its digits.
    - Repeat the process until the number equals 1 (where it will stay), or it loops endlessly in a cycle which does not include 1.
    - Those numbers for which this process ends in 1 are happy.
- Return true if n is a happy number, and false if not.



Example

Input: n = 19

Output: true

Explanation:

$1^2 + 9^2 = 82$

$8^2 + 2^2 = 68$

$6^2 + 8^2 = 100$

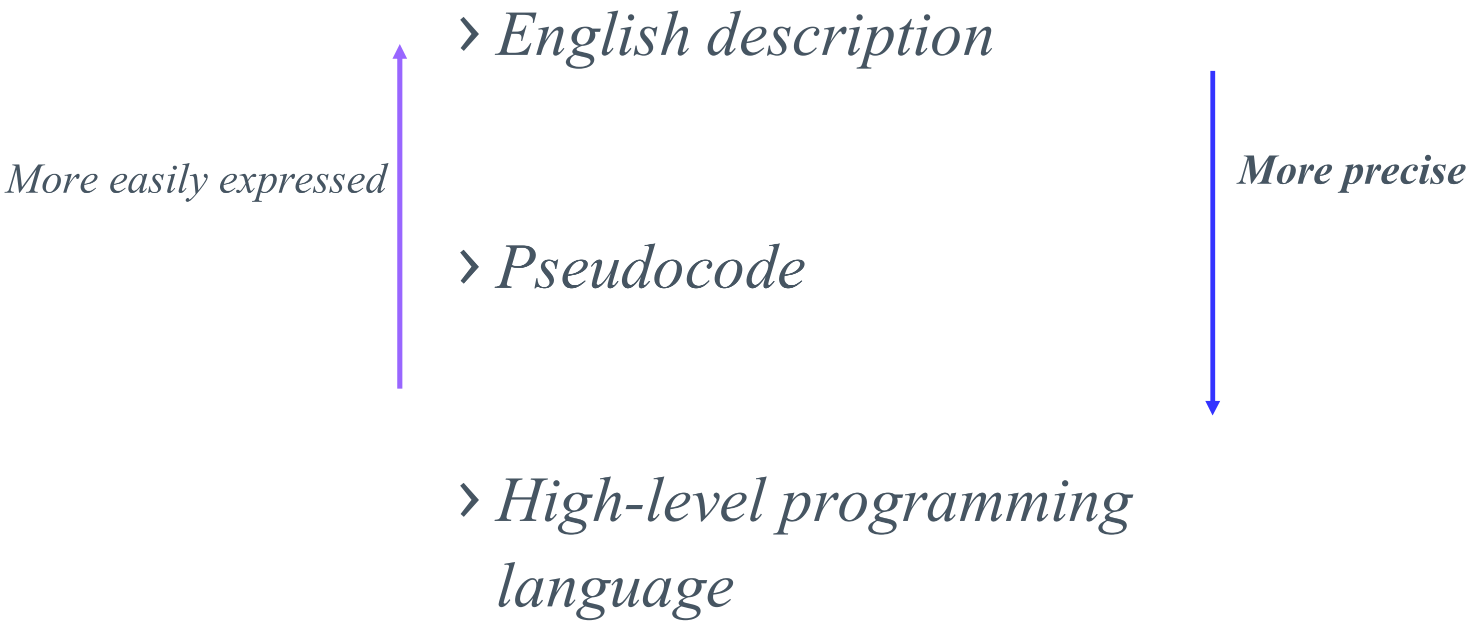
$1^2 + 0^2 + 0^2 = 1$

Test it for 7, 8, 9

Input: n = 2

Output: false

Expressing Algorithms



- *English description*
- *Pseudocode*
- *High-level programming language*

More easily expressed

More precise

Analysis of algorithms

- *Why analyze algorithms?*
  - *evaluate algorithm performance*
  - *compare different algorithms*
- *Analyze what about them?*
  - *running time, memory usage, solution quality*
  - *worst-case and "typical" case*
- *Computational complexity*
  - *Classifying problems according to difficulty level*
  - *algorithms provide upper bound scenario*

Analysis of algorithms (Cont !!!)

- *to show problem is hard or complete*
- *It requires at least a given amount of resources*
- *Transform problems to establish “equivalent” difficulty*

What is the "best" Algorithm

- *You can consider an algorithm best on the base answers to following questions.*
  - *How fast does it run?*
    - *Refer to processing time*
  - *How "complicated" is the algorithm*
    - *Time Complexity*
    - *Space Complexity*
  - *How well is the algorithm documented*
    - *Written pseudo code should be clear , complete and well documented*
  - *Can the machine used have influence on the results*
    - *Yes*
    - *Some good algorithms can not perform well on slow machines.*

Important Point to note

- *Programs depend on the operating systems, machine, compiler/interpreter used, etc.*
- *Analysis of algorithms compare algorithms and not programs.*
- *Performance of Algorithms should be measured before its implementation as program in any language.*
- *Algorithms should be checked on different machines.*

Example-1.

- *Consider following four pseudo codes (in C)*
- *Purpose of all pseudo codes is same.*
- *We are analysing them on the base do time and space trade-off factors*

PesuedoCode-1.

```
main()
{ int a, b, c;
  a=2; b=3;
  c=a + b;
  printf("%d", c);
}
```

Facts:

1. *Three variables (6 bytes)*
2. *Three assignment process*
3. *One Calculation*
4. *One output statement*

PesuedoCode-2.

```
main()
{
   int a, b;
   a=2;
   b=3;
   printf(“%d”, a+b);
}
```

Facts:

1. *Two variables (4 bytes)*
2. *Two assignment process*
3. *One Calculation*
4. *One output statement*

PesuedoCode-3.

```
main()
{
   int a,b,c;
   scanf(“%d%d”, &a, &b);
   c=a+b;
   printf(“%d”, c);
}
```

Facts:

1. *Three variables (6 bytes)*
2. *One input statement*
3. *One Assignment*
4. *One Calculation*
5. *One output statement*

PesuedoCode-4.

```
main()
{
   int a,b;
   scanf(“%d%d”, &a, &b);
   printf(“%d”, a+b);
}
```

Facts:

1. *Two variables (4 bytes)*
2. *One input statement*
3. *One Calculation*
4. *One output statement*

Comparison of Pseudo codes

Facts	Pseudo 1	Pseudo 2	Pseudo 3	Pseudo 4
Variables	3 (6 bytes)	2(4 bytes)	3 (6 bytes)	2 (4bytes)
No of Assignments	3	2	1	0
No of Calculation	1	1	1	1
Input Statements	0	0	1	1
Output Statements	1	1	1	1

Which one Pseudo code is best and how

- *With respect to space trade of Pseudo code 2 and Pseudo code 4 are candidate for best because in these pseudo codes 2 variables are used.*

- *But when focus on time tradeoff/complexity then Pseudo 2 code will best*

- *Why not Pseudo code 4 is best though, you are entering dynamic data (through scanf)*
  - *Because Some instruction access by microprocessor and some are executed by IO circuit of computer system.*
  - *Switching between IO and Microprocessor takes extra time*

Example-2. Pseudo code about finding swapping the position of two digits number

```
main()
{
  int N, a, b;
  N=54;
   a= N/10;
   b=N%10;
   N = b*10+a;
  printf(“%d”, N);
}
```

Facts:

1. *variables ?*
2. *input statement ?*
3. *Calculation ?*
4. *output statement ?*
5. *Assignments ?*

Example-2. Pseudo code about finding swapping the position of two digits number (Cont!!!)

Facts:

1. variables ? 3 (6 bytes)

2. input statement ? 0

3. Calculation ? 3

4. output statement ? 1

5. Assignments ? 3

Example-2. Pseudo code about finding swapping the position of two digits number (Cont!!!) (Second Pseudo code)

```
main()

{

   int N=54;

    N = (N%10)*10 + (N/10)

   printf(“%d”, N);

}
```

Facts:

1. *variables ?*
2. *input statement ?*
3. *Calculation ?*
4. *output statement ?*
5. *Assignments ?*

Example-2. Pseudo code about finding swapping the position of two digits number (Cont!!!)

Facts:

1. *variables ? 1 (2 bytes)*
2. *input statement ? 0*
3. *Calculation ? 1*
4. *output statement ? 1*
5. *Assignments ? 1*

Homework

- *Write at least three pseudo codes to find the largest of three numbers and analyzed these pseudo codes on base of space and time trade off factors.*

Summary

- *An algorithm or pseudo code will be considered best if it fulfill the time and space trade off/complexities.*
- *Performance of algorithm should be measured and not of implemented program.*
- *During writing algorithm or pseudo code, you must focus over the extra use of data structure and switching of instructions form CPU to IO circuits and vice versa*

Thank You!!!

Have a good day