Lecture 10

Software Complexity (Structural)

Software Structural Complexity

- What is software structural complexity ?
  - Estimated by physical lines of code (for any language)
  - How many variables, constants are there
  - Halstead's theory of measurement of software complexity :
  - Set of primitive measures that may be derived after code is generated or estimated once design is complete
  - Halstead uses the primitive measures to develop expression for

Software Structural Complexity

1. Overall program length
2. Program volume [critical volume/minimum volume for an algorithm, unit → number of bits]
3. Program level (a measure of software complexity)
4. Program effort (development effort)
5. Program time (development time)

Software Structural Complexity

- Parameters,
  - η1 =Total number of distinct/unique operators
  - η2 =Total number of distinct/unique operands
  - N1= Total number of all operators
  - N2= Total number of all operands
- Program length, N=N1+N2
- Operands = Variables and Constants
- Operators = Remaining all are belongs to operators

Software Structural Complexity

- By Halstead,
  1. Estimated program length,
  2. Program Volume, $N = \eta 1 \log_2 \eta 1 \quad \eta 2 \log_2 \eta 2$ bits
  3. Critical Volume, $V = (N1 + N2) \log_2 (\eta 1 + \eta 2)$ bits

- Can not able to create a program/algorithm/task less than 2 distinct operators and at least 2 distinct operands

Ex . Y=Φ(x) operator only for computation and one for assignment

$V^* = (2 + \eta 2^*) \log_2 (2 + \eta 2^*)$

= distinct number of actual i/p & o/p operands

$\eta 2^*$

Software Structural Complexity

4. Program Level, L=V*/V

5. Program Effort, E= V/L bits

6. Program Speed, S = E/s seconds (where s(mental discrimination) lies between 0 to 20)

Example : Include <stdio.h>

```
main() {

    int a,b,c;

    scanf("%d %d", &a, &b);

     c= a+b;

    printf("%d", c);   }
```