به نام خدا

شیرین حاجی امین شیرازی 9131009 تمرین 1

---

**2)**

**1-parentOf(john ,Mary)**

**2-parentOf(kay,john)**

**3-parentOf(bill,kay)**

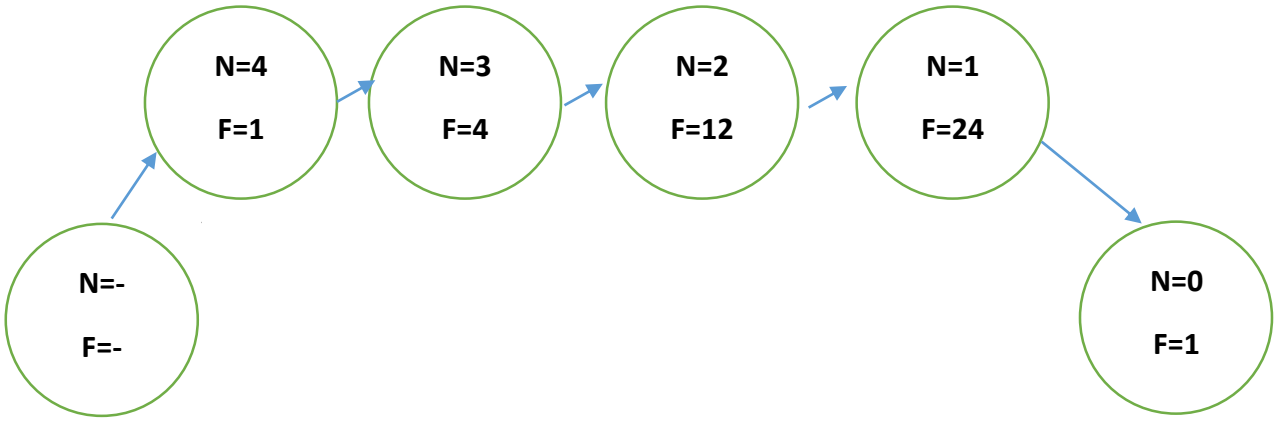
**4-ancestorOf(x,y) if parentOf(x,y)**

**5-ancestorOf(x,z) if parentOf(x,y) and ancestorOf(y,z)**

---

| | |
|---|---|
| **6-not ancestorOf(bill,mary)** | **assumption** |
| **7-x=bill** | **unification** |
| **8-z=mary** | **unification** |
| **9-not parentOf(x,y) or not ancestorOf(y,z)** | **modus tollens** |
| **10-y=john** | **unification** |
| **11-ancestorOf(john,mary)** | **(1),(4)** |
| **12-ancestorOf(bill,mary)** | **(11),(2),(5)** |
| **Contradiction** | **(6),(12)** |

(4




(5

1-concat([],l,l)

2-concat([x|l0],l1,[x|l2]) if concat(l0,l1,l2)

3-not concat([0,1],[a,b],l)

---

4-not concat([1],[a,b],l3) modus tolen

5-not concat([],[a,b],l4) modus tolen

6-not l4=[a,b] (1),(5)

7-not l3=[1,a,b] (4),(3)

8-not l=[0,1,a,b]

Concat([0,1],[a,b],[0,1,a,b])

(6

Imperative: c, basic,pascal,Ada,cobol,fortran,icon

Functional: haskell,lisp,APL,snobol

Logic:prolog,snobol

(7

imperative:teller machine,industrial robot

functional:flight control ,nuclear power station monitoring

logic:a legal advice service

(10

در زبان c ساختار های local بسیار بیشتر از ساختار های global هستند . برای مثال تعریف توابع ، متغییر ها و ... در متن برنامه به صورت local در نظر گرفته می شود و ساختار هایی مانند define ها global هستند و نمی توان آن ها را بر اساس flow در بلوک ها تغییر داد .

(13

اصول زبان های برنامه سازی به شرح زیر است :

سادگی، قانونمند بودن،فراهم کردن مکانیسم های انتزاع که امکان نوشتن و چک کردن کد و استفاده ی چند باره از آن را در اختیار بگذارد، ساختار های در معرض ارور نداشته باشد ،بتواند مدل اپلیکیشن را به بهترین نحو توصیف کند،برنامه را بدون وابستگی به محیط زیرین ( underlying environment) توصیف نماید .

زبان C که یکی از اصلی ترین زبان های imperative است ، از نظر سادگی بسیار قوی است ، ساختار های در معرض خطا ندارد ، قانونمند است اما در مقایسه با زبان جاوا که آن هم یکی از قدرتمند ترین زبانهای imperative است، قدرت انتزاع بسیار کمی را دارد ، کد ها در جاوا بسیار reusable هستند واستقلال کد بدون توجه به ساختار لایه ی زیرین بسیار بالاست . اما به طبع آن پیچیدگی بیشتری نسبت به c دارد .

(14

(b

```
public class MySort {

   public static void main(String[] args) {
```

```
    int[] input getInp();
    insertionSort(input);
  }
   Interface func{
      Public Boolean lessThan(a,b){
            If a<b
                  Return true;
            Return false;
            }
     }

  public static void insertionSort(int array[],func f) {
    int n = array.length;
    for (int j = 1; j < n; j++) {
      int key = array[j];
      int i = j-1;
      while ( (i > -1) && ( f.lessThan(key,array [i]) )) {
        array [i+1] = array [i];
        i--;
      }
      array[i+1] = key;
      printNumbers(array);
    }
  }
}
```