1. Describe the expression \((a - b * c) * (c * d - e) + f - (g - h)\) in Lisp's prefix form.

\[-(-(*(-a(+bc))((-cd)e))f)(-gh))\]
2. Use bottom-up algorithm to parse the expression from Problem 1, build the dependency graph, explain which operations can be performed in parallel first, which next, etc.

\[(a \cdot b \cdot c) \cdot (c \cdot d - e) + f - (g - h)\]

\[
\begin{align*}
  r_1 &\leftarrow a \cdot b \cdot c \\
  r_2 &\leftarrow a \\
  r_3 &\leftarrow c \cdot d \\
  r_4 &\leftarrow r_3 \cdot e \\
  r_5 &\leftarrow r_2 \cdot r_3 \\
  r_6 &\leftarrow r_5 \cdot f \\
  r_7 &\leftarrow -g \cdot h \\
  r_8 &\leftarrow r_6 \cdot r_7
\end{align*}
\]
3. Describe what will be the quadruples generated by Java based on the following program segment:

```java
for(int i = 1; i <= n; i++)
    {c[i] = a[i] + b[i];}

```

array c is all components of a added to all components of b starting at the second component.

```
L    r, <= i n
    if(r) else goto L2

    c[i] += a[i] b[i]

    i++
    goto L

L2
```

yes, the outcome of one iteration would have no effect on any other iteration no matter what order, since the c array is the one being changed and it's sources a and b are never changed.
5. Use the algorithm from Problem 3 to write a generic Java method for computing the component-wise sum of two arrays. This method should work for all possible numerical data types: int, short, char, double, float.

```java
public static <T> T[] sum2Array(<T> a, <T> b)
{
    int n;
    if (a.length > b.length)
    n = a.length - 1;
    else
    n = b.length - 1;

    <T>[] c = new <T>[n];
    c[0] = a[0] + b[0];

    for (int i = 1; i < n; i++)
    { c[i] = a[i] + b[i]; }

    return c;
}
```
6. Write a LISP function for computing the component-wise sum of two lists. Trace it on the example of the lists '(1 10 100) and '(2 20 200), the result should be '(3 30 300).

```
(defun listSum (list, list2)
  (cond
    ([nil list] list2)
    ([nil list2] list1)
    (t (cons (+ (car list) (car list2))
          (listSum (cdr list) (cdr list2))))
  )
)

(listSum '(1 10 100) '(2 20 200))
(cons (+12) (listSum '(10 100) '(20 200)))
(cons 3 (cons (+1020) (listSum '(100) '200)))
(cons 3 (cons 30 (cons (+100 200) (listSum '(1) '(1)))))
(cons 3 (cons 30 (cons 300 ')))
(cons 3 (cons 30 (cons 300)))
(cons 3 '(30 300))
'(3 30 300)
```
7. Write a Prolog program for computing the component-wise sum of the two lists. Trace it on the same example as in Problem 6.

\[
\text{listSum}([3,3,3])
\]

\[
\text{listSum}([H_1,T_1], [H_2,T_2], [H,T])
\]

\[
H \text{ is } H_1 + H_2
\]

\[
T \text{ is } [\text{Head}_2, \text{Tail}_1]
\]

\[
\text{listSum}([T_1], [T_2], [\text{Head}_2, \text{Tail}_2])
\]

\[
\text{listSum}([1,10,100], [2,20,200], [\text{Head}, \text{Tail}])
\]

\[
\text{head} = 1 + 2
\]

\[
\text{tail} = [30,300]
\]

\[
\text{listSum}([10,100], [20,200], [\text{Head}_2, \text{Tail}_2])
\]

\[
\text{head}_2 = 10 + 20
\]

\[
\text{tail}_2 = [300]
\]

\[
\text{listSum}([100], [200], [\text{Head}_3, \text{Tail}_3])
\]

\[
\text{head}_3 = 100 + 200
\]

\[
\text{tail}_3 = []
\]

\[
\text{listSum}([], [], [])
\]
8. We all know how to find the largest element of an array. Use Algol's call-by-name feature to write a method \texttt{max(index, lower, upper, expression)} that would be able to compute the largest value of a given expression when the index of this expression ranges between the given lower bound and the given upper bound. Show, step by step, how this method can be used to find the largest value of the expression \((0.1 \times j) - \sqrt{(0.1 \times j)}\), when \(j\) ranges from 0 to 10.

\[
\begin{align*}
\text{max}(\text{index}, \text{lower}, \text{upper}, \text{expression}) & \equiv \\
\text{index} &= \text{lower}; \\
\text{double max} &= \text{expression}; \\
\text{for}(\text{index}; \text{index} \leq \text{upper}; \text{index}++) & \quad \text{if}(\text{max} < \text{expression}) \\
\quad & \text{max} = \text{expression}; \\
\text{return max}
\end{align*}
\]

\[
\begin{align*}
\text{max}(0, 0, 10, (0.1 \times j) - \sqrt{(0.1 \times j)}) & \equiv \\
\text{j=0} & \quad \text{max=0} \\
\text{double max} &= (0.1 \times j) - \sqrt{(0.1 \times j)} \\
\text{for}(\text{j}; \text{j} \leq 10; \text{j}++) & \quad \text{if}(\text{max} < ((0.1 \times j) - \sqrt{(0.1 \times j)})) \\
\quad & \text{max} = (0.1 \times j) - \sqrt{(0.1 \times j)} \\
\text{return max}
\end{align*}
\]

\[
\begin{align*}
\text{j=0} & \quad \text{max=0} \\
\text{j=1} & \quad \text{if}(0 < 0.1 - \sqrt{0.1}) \times 0 \quad \text{j=2} & \quad \text{if}(0 < 0.2 - \sqrt{0.2}) \times 0 \\
\quad & \text{max=0} \\
\text{j=2} & \quad \text{if}(0 < 0.1 - \sqrt{0.1}) \times 0 \quad \text{j=2} & \quad \text{max=0}
\end{align*}
\]
9. Use the predicates parent, male, and female to describe the concept of a brother. Test your definition on the following database related to the Russian czar Peter I, the founder of St. Petersburg, Russia:

parent(alexei, peter).
parent(alexei, ivan).
parent(alexei, sofia).
male(peter).
male(ivan).
female(sofia).
\text{? brother}(X, Y).

\text{brother}(X, Y) \leftarrow \text{male}(X), \text{parent}(p, X), \text{parent}(p, Y), X \neq Y

\text{brother}(x, y) \leftarrow \text{male}(\text{peter}), x = \text{peter}
\text{parent}(\text{alexei, peter}) p = \text{alexei}
\text{parent}(\text{alexei, peter}) x \text{ peter } y
\text{parent}(\text{alexei, ivan}) y = \text{ivan} \leftarrow \text{brother}(\text{peter, ivan})
\text{or}
\text{parent}(\text{alexei, sofia}) y = \text{sofia} \leftarrow \text{brother}(\text{peter, sofia})

\text{male}(\text{ivan}), x = \text{ivan}
\text{parent}(\text{alexei, ivan}) p = \text{alexei}
\text{parent}(\text{alexei, peter}) y = \text{peter} \leftarrow \text{brother}(\text{ivan, peter})
\text{parent}(\text{alexei, ivan}) y \neq \text{ivan}
\text{parent}(\text{alexei, sofia}) y = \text{sofia} \leftarrow \text{brother}(\text{ivan, sofia})
10. Use wave algorithm to solve the following problem:

- we know that distance $d$ is equal to velocity $v$ multiplied by time $t$: $d = v \times t$;
- we know that the cost $c$ of the gas is proportional to the amount of fuel: $c = 2.10 \times f$;
- we know that the amount of fuel $f$ is proportional to the distance: $f = d / 20$.

We know the velocity $v$ and the time $t$. We need to compute the cost $c$ of the gas.