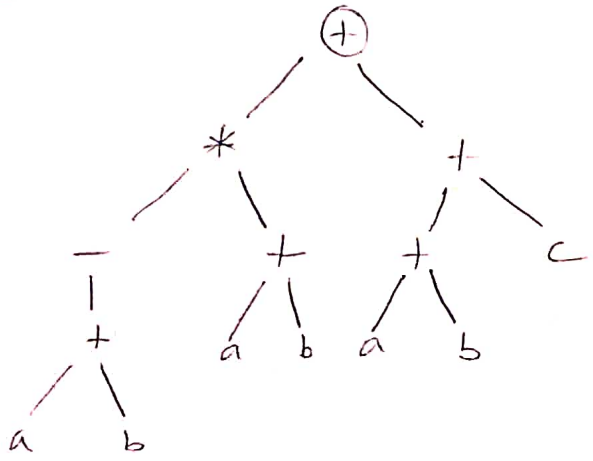
Name : SWARNAVA CHAKRABORTY
Roll no. 20, Dept. : CSE , Year : 4th
Subject : Compiler Design Assignment

Date : 22-02-2021

(1) Translate the expression :- $(a+b)*(c+d)+(a+b+c)$ into –

a) Syntax tree
b) Quadruples
c) Triples
d) Indirect Triples

a) Syntax tree :



b) Quadruples :-

Location	operator	Argument 1	Argument 2	Result
0	+	a	b	T_1
1	−	T1		T_2
2	+	c	d	T_3
3	*	T2	T_3	T_4
4	+	T1	c	T5
5	+	T4	T5	T6

c) Triples

Location	Operator	Argument 1	Argument 2
0	+	a	b
1	−	(0)	
2	+	c	d
3	*	(1)	(2)
4	+	(0)	c
5	+	(3)	(4)

d) Indirect Triples

#	statement
0	(14)
1	(15)
2	(16)
3	(17)
4	(18)
5	(19)

Location	Operator	Argument 1	Argument 2
(14)	+	a	b
(15)	−	(14)	
(16)	+	c	d
(17)	*	(15)	(16)
(18)	+	(14)	c
(19)	+	(17)	(18)

(2) Translate the executable statement of the following C program into three-address code.

```
main( ) {
    int i;
    int a[10];
    i=1;
    while (i<=10) {
        a[i] = 0; i=i+1; }
}
```

Three - address Code:

```
i = 1;
L : T1 = 0
    T2 = &a
    T3 = sizeof(int)
    T4 = T3 * i
    T5 = T2 + T4
  * T5 = T1
    i = i + 1
if  i ≤ 10 goto L
```

(3) Ans:- 3) a)

f(5)	f(5), 5, s = f(4), t = f(3)
f(4)	f(4), 4, s = f(3), t = f(2)
f(3)	f(3), 3, s = f(2), t = f(1)
f(2)	f(2), 2, s = f(1), t = f(0)
f(1)	f(1), 1

3) b)

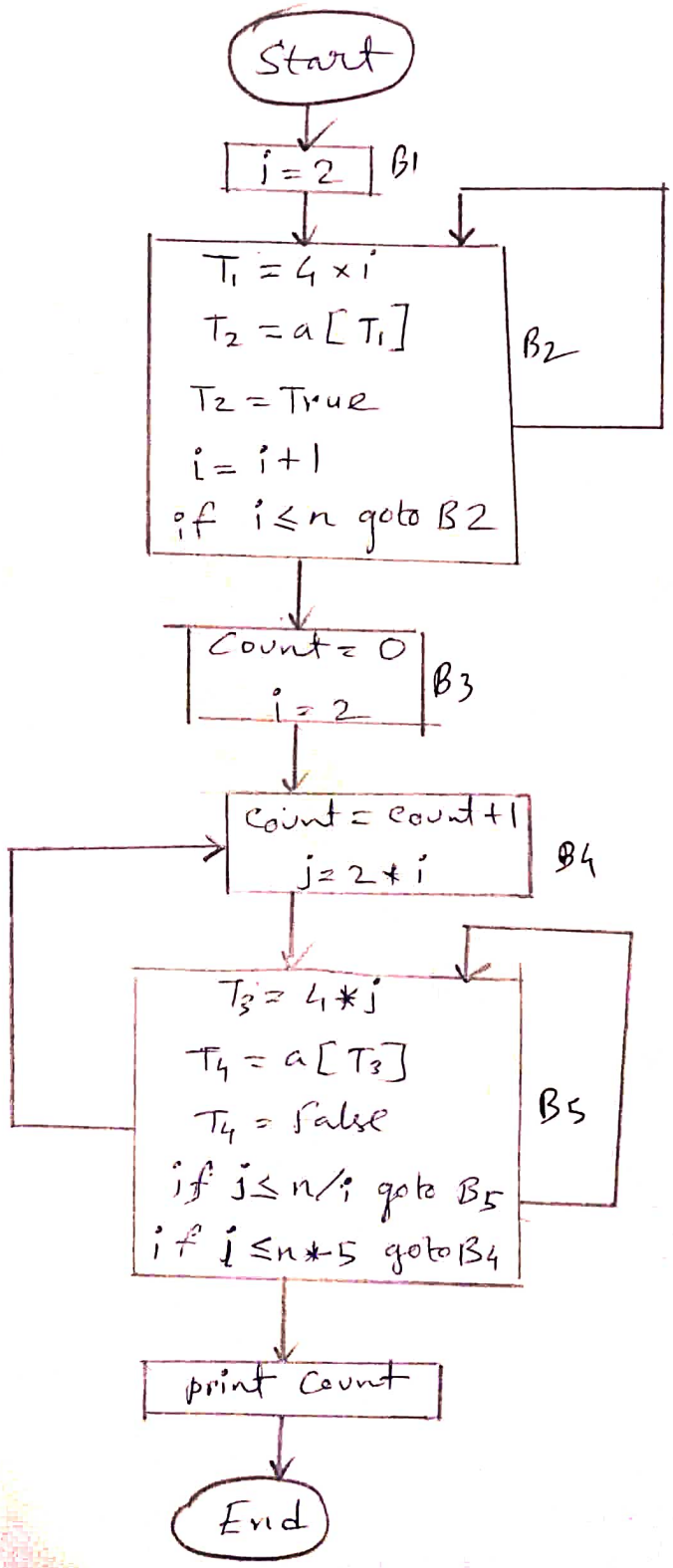
f(5), 5, s = f(4), t = f(3)
f(3), 3, s = f(2), t = f(1)
f(1), 1

4) Ans:—

4) a) Three address code:

```
    i = 2
L1 = T1 = true
     T2 = &a
     T3 = sizeof(int)
     T4 = T3 * i
     T5 = T2 + T4
    *T5 = T1
     i = i + 1
  if i<=n  goto L1
   count = 0
   i=2
L2: count = count + 1
    j = 2 * i
L3: T6 = false
    T7 = T3 * j
    T8 = T2 + T7
   *T8 = T6
  if j<=n/i goto L3.
   if i<=n*5 goto L2.
```

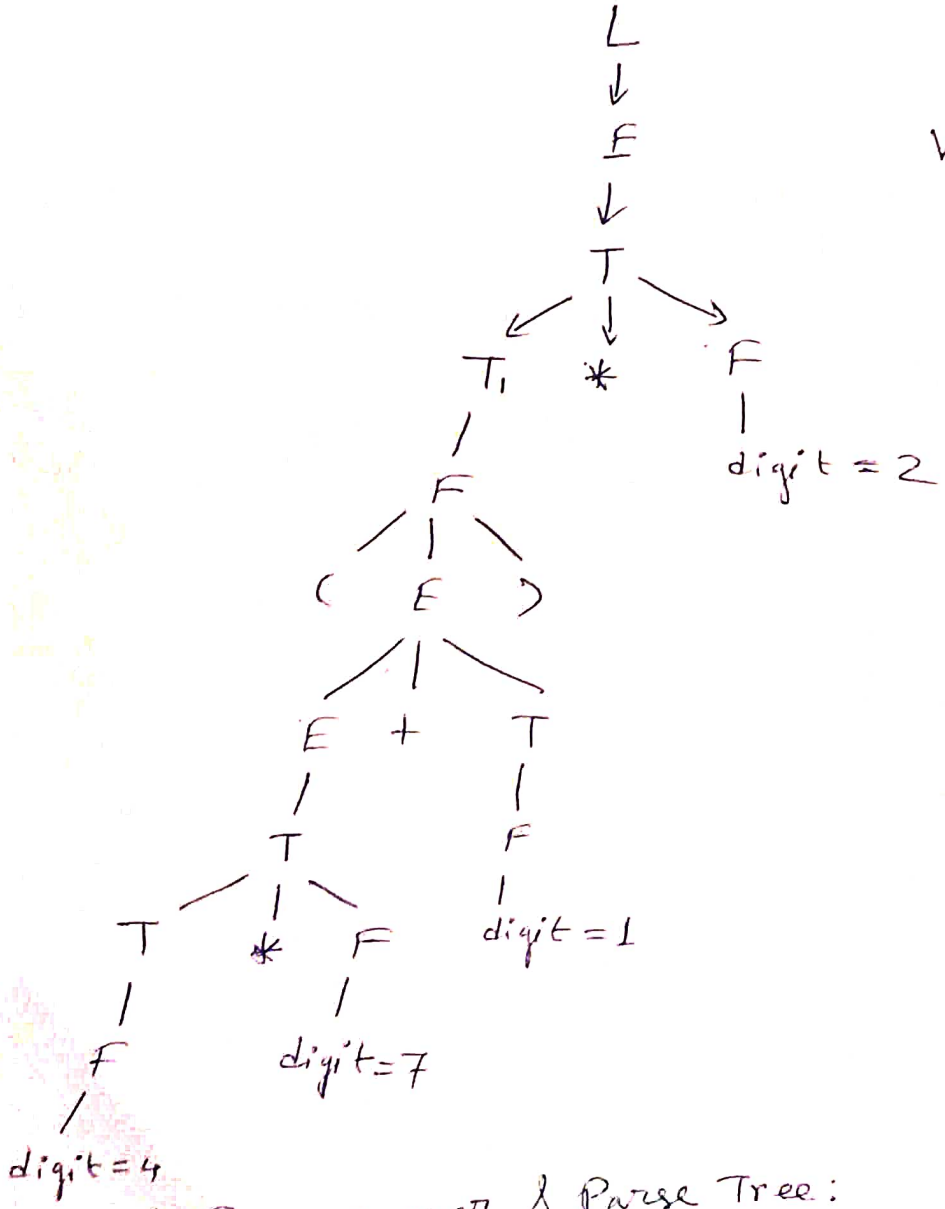
4) b) Flow graph :



5) Ans :-

Expression : $(4 * 7 + 1) * 2$

- Annotated Parse Tree :-



Value = 58

- Context free grammar & Parse Tree :

Production	Semantic Action
$L \to E$	L.val = E.val.
$E \to E1 + T$	E.val = E1.val + T.val
$E \to T$	E.val = T.val
$T \to T1 * F$	T = T1.val * F.val
$T \to F$	T.val = F.val
$F \to (E)$	F.val = (F.val)
$F \to digit$	F.val = Digit

Now,

$L \to E$

$E \to E + T / T$

$T \to T * F / F$

$F \to (E) / digit$

First

first(L) = (, digit

First(E) = (, digit

first(T) = (, digit

first(F) = (, digit

Follow

follow(L) = $

follow(E) =), T, $

follow(T) =), *, T, $

follow(F) =), *, +, $

Parse Table: (LL(1))

	(	)	digit	+	*	$
L	L→E		L→E			
E	E→E+T E→T		E→E+T E→T			
T	T→T*F T→F		T→T*F T→F			
F	F→(E)		F→digit			