


**VIT**

Vellore Institute of Technology

## Final Assessment Test – November 2024

Course: BCSE307L - Compiler Design

Class NBR(s): 1542/1548/1555/1587/1605/1612/

1623/1633/1641/1651/1660/1669/1673/1676/1684/ Slot: C1+TC1

1725/1734/1740/1746/7999

Time: Three Hours

Max. Marks: 100

- KEEPING MOBILE PHONE/ANY ELECTRONIC GADGETS, EVEN IN 'OFF' POSITION IS TREATED AS EXAM MALPRACTICE
- DON'T WRITE ANYTHING ON THE QUESTION PAPER

Answer ALL Questions

(10 X 10 = 100 Marks)

1. Explain the structure of a compiler. Illustrate the output of each phase of compilation for the given input:

```
int factorial(int n)
{
    if(n==0)
        return 1
    else
        return n*factorial(n-1);
}
```

2. Check whether the following grammar is LALR(1) or not and give suitable justification.

 $S' \rightarrow S$ 
 $S \rightarrow aBc \mid bCc \mid aCd \mid bBd$ 
 $B \rightarrow e$ 
 $C \rightarrow e$ 

3. Define Backpatching semantic actions the following Grammar:

```
E → E1 OR M E2
    | E1 AND M E2
    | not E1
    | ( E1 )
    | id1 relop id2
    | true
    | False
M → ε
```

Based on the translation scheme, generate intermediate code representation for the following expression.

a)  $a==b \text{ AND } (c==d \text{ OR } e==f)$ 

b)  $(a==b \text{ AND } c==d) \text{ AND } e==f$

4. Generate 3AC for the following code snippet and represent the following code into Quadruple, triples and indirect triples form.

```
main( )
{
    int i;
    int a[10];
    i = 1;
    while (i <= 701 )
    {
        a[i] = 0;
        i = i + 1;
    }
}
```

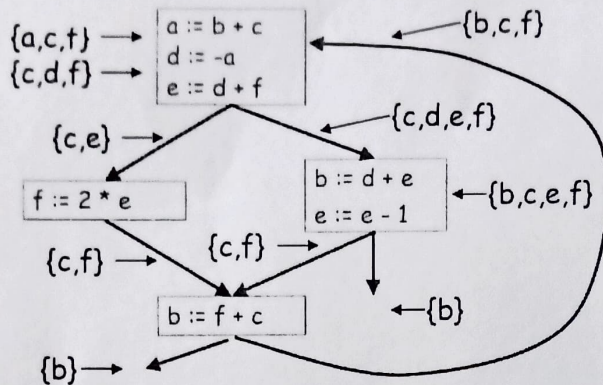
5. Identify the basic blocks and construct a flow graph for the below prime number counting code from a large given array. Also identify loops in the flow graph.

```
for (i=2; i<=n; i++)
    a[i] = TRUE;
count = 0;
s = sqrt(n);
for (i=2; i<=s; i++)
    if (a[i]) /* i has been found to be a prime */
    {
        count++;
        for (j=2*i; j<=n; j = j+i)
            a[j] = FALSE; /* no multiple of i is a prime */
    }
```

6. Give the DAG representation of basic block for the following three address code and apply suitable peephole optimization techniques to generate optimized code.

```
(1) t1 := 4*i
(2) t2 := a [t1]
(3) t3 := 4*i
(4) t4 :=b [t3]
(5) t5 := t2*t4
(6) t6 := prod +t5
(7) prod := t6
(8) t7 := i+1
(9) t8 :=4*2
(10) if t8 <5 goto (4)
(11) i := t7
(12) If i<=20 goto (1)
```

7. Explain register allocation using graph coloring for the below flow graph code sequence.



8. Explain the role of Instruction Scheduling and Software Pipelining in compiler design with suitable examples and give its significance.

- 9.a) Consider the following grammar

```

P → S
P → S P
S → if E then S
S → if E then S else S
S → while E S
S → begin P end
S → print E
S → E
E → id
E → integer
E → E + E
  
```

- Write a new grammar which accepts the same language but avoids left recursion and common left prefixes and figure out the FIRST and FOLLOW sets for the new grammar.
- Construct the LL (1) parse table for the new grammar; is the new grammar is LL (1) grammar? Explain your answer carefully.

OR

- 9.b) Construct a recursive descent parser for the grammar  $S \rightarrow '(S)' \mid \epsilon$ . Will it parse correctly?

- 10.a) The following grammar generates expressions formed by applying an arithmetic operator + to integer and real constants. When two integers are added, the resulting type is integer, otherwise, it is real

$$E \rightarrow E+T \mid T$$

$$T \rightarrow \text{num.num} \mid \text{num}$$

- i. Give a syntax-directed definition to determine the type of each sub expression.
- ii. Extend the syntax-directed definition of (a) to translate expressions into postfix notation as well as determining types. Use the unary operator 'inttoreal' to convert an integer value into an equivalent real value, so that both operands of + in the postfix form have the same type.

OR

- 10.b) Formulate SDT for syntax tree construction for a simple desk calculator and show how the expression  $2+4*5^2$  is evaluated.

BI/K/TX