📘 Compiler Design UNIT–IV PYQ Analysis

🔥 Intermediate Code Generation & Code Generation

🎯 RGPV PYQ Analysis (2017–2020)

Maine uploaded PYQ papers analyze kiye aur UNIT–IV se related sabhi important questions extract kiye.
Ye unit RGPV me:

- Numerical based
- DAG based
- Optimization based

  questions ke liye bohot important hai.

🌟 UNIT–IV Topics

📌 Intermediate Code Generation

- Three Address Code
- Quadruples
- Triples
- Boolean Expressions
- Backpatching
- Procedure Calls

📌 Code Generation

- Basic Block
- Flow Graph
- DAG Representation
- Register Allocation
- Peephole Optimization
- Code Generation from DAG

🔥 MOST REPEATED TOPICS

Topic	Frequency	Importance
DAG Representation	⭐⭐⭐⭐⭐	VERY HIGH
Intermediate Code Generation	⭐⭐⭐⭐⭐	VERY HIGH
Quadruples & Triples	⭐⭐⭐⭐	HIGH
Peephole Optimization	⭐⭐⭐⭐	HIGH
Basic Block	⭐⭐⭐⭐	HIGH
Optimization Techniques	⭐⭐⭐⭐	HIGH
Operator Precedence Parser	⭐⭐⭐	Medium
Backpatching	⭐⭐⭐	Medium-High

📅 YEAR-WISE PYQ ANALYSIS

🔥 DECEMBER 2020 PAPER

✅ Q7(b)

How does an Operator Precedence parser work? Use a pre-constructed operator precedence table to guide the parsing of input:

```
a+b-20
```

📌 Related Topic:

- Operator Precedence Parsing
- Intermediate Code Concepts

⭐ IMPORTANT

🔥 JUNE 2020 PAPER

✅ Q1

Construct a DAG for the basic block whose code is given below:

```
D = B * C
E = A + B
B = B * C
A = E - D
```

📌 Related Topic:

- DAG Representation
- Basic Block

⭐ MOST IMPORTANT

✅ Q5

How does an Operator Precedence parser work? Use a pre-constructed operator precedence table to guide parsing of:

```
a+b-20
```

📌 Related Topic:

- Operator Precedence Parsing

⭐ REPEATED

✅ Q6

Define a Quadruple.

How is it different from triples?

Convert the following expression into three address code:

```
c = (a+b)/(c-d) * (e+f)
```

📌 Related Topic:

- Intermediate Code Generation
- Three Address Code
- Quadruples & Triples

⭐ VERY IMPORTANT

✅ Q7

Construct DAG for the following basic block:

```
a = b + c
b = b - d
c = c + d
e = b + c
```

📌 Related Topic:

- DAG Representation

⭐ REPEATED AGAIN

✅ Q8(a)

Write short notes on:

- Local and Loop Optimization
- Peephole Optimization
- Dead Code Elimination

📌 Related Topic:

- Optimization Techniques
- Peephole Optimization

⭐ VERY IMPORTANT

🔥 DECEMBER 2017 PAPER

✅ Q6(a)

Translate the following code into three address code:

```
a = 10;
b = 6;

if(a>b+5)
   b = 5;
```

📌 Related Topic:

- Intermediate Code Generation
- Three Address Code

⭐ VERY IMPORTANT

✅ Q6(b)

What is DAG? Construct DAG for the basic block:

```
D = B * C
E = A + B
B = B * C
A = E - D
```

📌 Related Topic:

- DAG Representation

⭐ MOST REPEATED QUESTION

✅ Q7(a)

Analyse the possible causes of dead code. Explain with an example how compiler can detect presence of dead code.

📌 Related Topic:

- Dead Code Elimination
- Optimization

⭐ IMPORTANT

✅ Q7(b)

Explain the common subexpression elimination, copy propagation and transformation for moving loop invariant computations.

📌 Related Topic:

- Optimization Techniques
- Code Optimization

⭐ VERY IMPORTANT

🔥 FINAL ANALYSIS (MOST IMPORTANT QUESTIONS)

⭐ TOP 7 MOST IMPORTANT QUESTIONS

1️⃣ Construct DAG for a Basic Block

🔥 Asked multiple times

🔥 Highest probability

2️⃣ Explain Intermediate Code Generation

🔥 Frequently repeated

3️⃣ Explain Three Address Code, Quadruples & Triples

🔥 Very important numerical question

4️⃣ Explain Peephole Optimization

🔥 Repeated optimization topic

5️⃣ Explain Basic Block and Flow Graph

🔥 High chance theory question

6️⃣ Explain Dead Code Elimination

🔥 Repeated optimization topic

7️⃣ Explain Common Subexpression Elimination

🔥 Important optimization technique

🎯 MOST IMPORTANT SMALL TOPICS

Topic	Chance
Register Allocation	HIGH
Backpatching	HIGH
Boolean Expressions	HIGH
Procedure Calls	MEDIUM
Flow Graph	HIGH
Loop Optimization	HIGH
Copy Propagation	HIGH

🔥 MOST EXPECTED QUESTIONS FOR UPCOMING EXAM

⭐ VERY HIGH CHANCE

1. Construct DAG for a Basic Block.
2. Explain Intermediate Code Generation.
3. Explain Quadruples and Triples.
4. Explain Peephole Optimization.
5. Explain Basic Block and Flow Graph.

⭐ HIGH CHANCE

6. Explain Register Allocation and Assignment.
7. Explain Dead Code Elimination.
8. Explain Common Subexpression Elimination.
9. Explain Backpatching with example.

⭐ MEDIUM CHANCE

10. Explain Procedure Calls.

11. Explain Boolean Expressions in Intermediate Code.
12. Explain Issues in Design of Code Generator.

⚡ SMART STUDY STRATEGY FOR UNIT–IV

🌟 MUST STUDY FIRST

✅ DAG Representation
✅ Basic Block
✅ Three Address Code
✅ Quadruples & Triples

🌟 THEN STUDY

✅ Peephole Optimization
✅ Dead Code Elimination
✅ Common Subexpression Elimination

🌟 LAST

✅ Register Allocation
✅ Procedure Calls
✅ Backpatching

🔥 ONE NIGHT REVISION PRIORITY

1. DAG Representation
2. Three Address Code
3. Quadruples & Triples
4. Basic Block
5. Peephole Optimization
6. Dead Code Elimination
7. Common Subexpression Elimination
8. Register Allocation
9. Backpatching