Rebecca Dias
19 /182027
TE CMPN A

SPCC

Question 3 a.

→ Code Optimization Techniques.
i) Dead code elimination
ii) constant folding and propagation
iii) Variable propagation
iv) common sub-expression elimination
v) Algebric simplification and strength reduction
vi) loop optimization

A) Dead code elimination
① Portion of the program that is never visited and executed under any situation is called dead code
② It can be removed from the program without affecting behaviour and performance
eg:
```
int main(){
    ...
    flag = false;
    if (flag)
        a = b + c   ← dead code
    ...}
```

B) constant folding and propagation
① It refers to placement of expression

Rebecca Dias
19/182027

that can be evaluated at compile time by their computed values

② It corresponds to determination of an expression with constant operands and replacing it with a single value

eg:- a = 5+3 +b+c can be
a = 8+b+c

④ Name variables replaced with constants which are assigned to them called indirect optimization

eg:- PI = 3.14
R = PI/180 ⇒ R = 3.14/180
= 0.01

C) Variable Propagation

① Replaces one variable with another holding an identical value.

② May increase probability of generating common sub-expressions

eg:-
```
a=b                    a=b
...                    ...
y=a*4          →       y=b*4
z=b*4+10;              z=b*4+10;
...                    ...
```

D) Common sub-expression elimination

① Some expressions may contain a sub-expression which may compute same value

② common sub-expression can only be

Rebecca Dias
19/182027

eliminated if it computes the same value for 2 or more distinct expressions.

eg:-

```
. . .
. . .
y = a+b ;
. . .
z = a+b+10 ;
. . .
```

→

```
. . .
temp = a+b
y = temp
. . .
z = temp + 10 ;
. . .
```

E) Algebric simplification and strength Reduction

① Replacing simple algebric expressions by a simplified expression

example : $a = x \times 0 \longrightarrow a = 0$

$a = x \times 1 \longrightarrow a = x$

② Replacing expensive machine instructions by cheaper ones.

eg: $x^2 \longrightarrow x \times x$

$x * 2 \longrightarrow x << 1$

F) Using loops for code optimization

① code motion
② loop fission
③ loop interchange
④ loop peeling
⑤ loop unrolling