62020 SEM:06
Rebecca SPCC

BARFI Page No.
Date / /

6/10

Q3]
A]

ii]

Application software	System software
① Application software is used by user to perform specific task	① System software is used for operating computer software
② Application software are installed according to user's requirements	② System software are installed in the computer when operating system is installed
③ In general, the user interacts with application software	③ In general, the user does not interact with system software because it works in the background.
④ Application software can't run independently. They can't run without the presence of software system	④ System software can run independently. It provides platform for running application softwares
⑤ Some examples of application software are word processor, web browser, media player etc	⑤ Some examples of system software are compiler, assembler, debugger, driver etc.
⑥ Application software carries a specific purpose	⑥ System software is endowed with a general purpose.

G 2020 SEM: 06
Rebecca SPCC

BARFI Page No.
Date

7/10

Q3]
A]

i]
① A grammar is said to be left recursive if it has a non-terminal, say A such that there is a derivation $A \Rightarrow A\alpha$, for some string α

② The general algorithm to remove direct left recursion follows, discard any rules of the form $A \rightarrow A$ and consider those that remain. Replace these with two sets of production, one set for $A \rightarrow A\alpha \mid \beta_1$

$A \rightarrow \beta_1 A'$
$A' \rightarrow \alpha A' \mid \varepsilon$

and another set for A'

③ Repeat this process untill no direct left recursion remain

④ Consider the following example

$E \rightarrow E+T \mid T$
$T \rightarrow T*F \mid F$
$F \rightarrow (E) \mid id$

Elimination of left recursion from the rules modifies the grammar as,

$E \rightarrow TE'$
$E' \rightarrow +TE' \mid \varepsilon$
$T \rightarrow FT'$
$T' \rightarrow *FT' \mid \varepsilon$
$F \rightarrow (E) \mid id$

8/10

Q3]

B] Optimization is the process of transforming a piece of code to make more efficient without changing its output.

(i) Loop optimization techniques with eg.

Loop optimization has the following types :-

i) code motion
ii) loop unrolling
iii) loop interchange
iv) loop splitting
v) loop peeling
vi) loop fission.

(1) code motion

- Sometimes loop may contain expressions that retain its value (invariant expressions)
- Such expressions can be moved before the loop
- This reduces frequency of execution of the loop expression

eg:-

```
:
while (i< n-2)
{
:
}
```

→

```
t = n-2;
while (i<t)
{
:
}
```

(2) Loop unrolling

- Body of the loop is reproduced a number of times
- This method improves efficiency by reducing loop overhead - test and loop variable
- Loop structure may disappear

- In partial unrolling increases source statements but reduces loop execution

eg:-

```
for (int i=0 ; i<100; i++)
    {
    print ("%d \n", i);
    }
    :
```

⟶

```
for (i=0 ; i<100; i=i+5)
{ print ( )
  print ( )
  print ( )
  print ( )
} print ( )
```

③ Loop interchange
- Inner and outer loops are interchanged to improve locality of reference

eg:-

```
for (int j=0; j<50; j++)
    for (int i=0 ; i<50, i++)
        A[i][j] = ...;
    ...
```

⟶

```
for (i=0; i<50; i++)
for (int j=0; j<50; j++)
    A[i][j] = ...;
```

④ loop splitting
- Breaking of a single loop into multiple loops

eg:-

```
for i=1 to 100 do
    if i<50 then b1
    if 50<=i <60 then
        b2
    else
        b3
end loop
```

⟶

```
for i=1 to 49 do
    b1
end loop
for i=50 to 59 do
    b2
end loop
for i=60 to 100 do
    b3
end loop
```

G2020 SPCC
SEM 06
Rebecca

BARFI Page No.
Date / /

10/10

(5) loop fission
- Breaking a loop over multiple loops with same index range
eg:-

```
...
for i=1 to 50
    x[i]=0;
    y[i]=0;
end loop
...
```

→

```
...
for i=1 to 50
    x[i]=0;
end loop
for i=1 to 50
    y[i]=0;
end loop
```

(6) loop peeling
- If there is a condition on the first value of the loop variable, it can be performed seperately
eg:-

```
...
for i=1 to 50 do
    if i=1 then b1;
        else b2;
end loop
```

→

```
i=1
    b1;
for i=2 to 50 do
    b2;
end loop
...
```