1. На линију испред значења унети број којим је означен одговарајући допунски параметар:

1) (#) 2) (0) 3) (+) 4) (-)

_4_ Означава да ће се поравнање вршити уз леву ивицу поља ширине n знакова, допунски знакови размака додају се иза, а не испред податка

_3_ Означава да се испред позитивног броја мора исписати знак плус

_2_ Нула код нумеричких података означава да ће се приликом равнања уз десну ивицу број допуњавати нулама, а не знаковима размак

_1_ Исписује се децимална тачка у конверзији рационалних бројева који немају разломљени део

2. На линију испред вредности скретнице унети редни под којим је наведен одговарајући екрански приказ:

```
switch(c) {
case 'A': case 'a': printf("Pravougaonik "); case 'B': case 'b': printf("Trougao "); break;
case 'C': case 'c': printf("Krug "); default: printf("Duz "); break; }
```

1) Krug Duz
2) Pravougaonik Trougao Krug Duz
3) Krug
4) Trougao
5) Pravougaonik Trougao
6) Duz

_1_ 'c'
_5_ 'A'
_6_ 'K'
_4_ 'b'

РЕШЕЊЕ:
Уколико изоставимо кључну реч break унутар неког од case-ова програм ће се извршавати док не дође до првог наредног break-а.

3. На линију испред типа података унеси број којим је означена одговарајућа конверзија:

1) %d
2) %i
3) %s
4) %ld
5) %f
6) %e
7) %hd
8) %u

_4_ long
_8_ unsigned
_7_ short
_1_ signed int (u dekadnom obliku)
_2_ signed int (dekadni, heksadekadni ili oktalni oblik)

4. Са леве стране наведене су функције за читање и упис у датотеку, а са десне стране опис функције. Повезати их:

1) fprintf
2) fgetc
3) fputs
4) fscanf
5) fgets

_2_ учитавање карактера из датотеке
_5_ учитавање реда из датотеке
_1_ форматирани упис података у датотеку
_3_ упис стринга у датотеку
_4_ форматирано учитавање података из датотеке

5. На линију испред описа упишите број под којим је наведена одговарајућа escape секвенца:

1) '\t'
2) '\n'
3) '\b'
4) '\r'
5) '\h'
6) '\a'

_4_ враћање на почетак реда (carrage return)
_6_ системски звучник (bell)
_2_ прелаз у нови ред (new line)
_5_ није escape секвенца
_1_ хоризонтални табулатор (horizontal tab)
_3_ враћање једну курсорску позицију назад (backspace)

6. Проценити који од наредних исказа који дефинишу дату службену реч НИЈЕ тачан.

1) Службена реч base може послужити за позивање конструктора родитељске класе.
2) Службена реч base може послужити за позивање приватних метода родитељске класе којима се другачије не може приступити.
3) Службена реч base може послужити за позивање заклоњеног метода родитељске класе.
4) Службена реч base може послужити за позивање заклоњеног поља родитељске класе.

7. Анализирати дати код и проценити шта ће се догодити након његовог извршавања:

```
static void Main(string[] args){
int[] a = new int[5];
for (int i = 0; i < a.Length; i++) a[i] = i;
Console.Write(a[i] + " "); }
```

1) Програм има грешку, јер променљива i у последњој наредби Console.Write у методу Main неће имати дефинисану вредност.
2) Програм приказује бројеве 0 1 2 3 4 на екрану.
3) Програм има грешку, јер ће у последњој наредби Console.Write метода Main покушати приступ непостојећем елементу a[5].
4) Програм приказује број 5 на екрану.

РЕШЕЊЕ:
Променљива i је дефинисана унутар for петље и постоји само унутар ње, односно до првог знака ; .

8. Анализирати код и одредити резултат који ће се приказати на екрану.

```
static void Main(string[] args) { fun(2); }
public static void fun(int n) { while (n > 1) { Console.Write((n - 1) + " "); fun(n - 1); } } } }
```

Програм на екрану не приказује ништа

1) Програм на екрану бесконачно приказује 1 1 1 1 1 ...
2) Програм на екрану бесконачно приказује 2 2 2 2 2 ...
3) Програм на екрану приказује 1 2 3.
4) Програм на екрану приказује 3 2 1.

РЕШЕЊЕ:
С обзиром да нигде не мењамо вредност параметра n он ће увек бити већи од 1 и бесконачан број пута ће исписивати 1. Позив функције fun(n-1) неће радити ништа јер 1 није веће од 1 и ту ће се завршити.

9. Заокружити број испред понуђеног тачног исказа:

```
class Test { int x;
public Test(string s){ Console.WriteLine("Klasa Test"); }
static void Main(string[] args){ Test t = null; Console.WriteLine(t.x); }}
```

1) Програм има грешку јер променљива x није иницијализована.
2) Програм има грешку јер класа Test нема подразумевани конструктор.
3) Програм има грешку јер се у некој класи не може декларисати променљива типа те исте класе, као што је то овде случај са променљивом t.
4) Програм има грешку јер променљива t није иницијализована и има вредност null у моменту када се приказује поље t.x .
5) Програм нема грешака и нормално се извршава, не приказујући ништа на екрану.

РЕШЕЊЕ:
Уколико нисмо креирали објекат не можемо ни приступати његовим пољима.

10. Анализирати дати код и одредити да ли је код исправно написан.

```
class Program {
 public static void Main(string[] args) {
 B b = new B();
 b.Metod(5);
 Console.WriteLine("b.i je " + b.CitajI()); } }
```

```
class A {
 int i;
 public int CitajI(){return i;}
 public void Metod(int i) { this.i = i; } }

class B : A {
 public void Metod(string s){
 Console.WriteLine(s); } }
```

1) Програм нема грешке, јер наслеђени метод класе A, Metod(int i) није надјачан у класи B, већ је дефинисан преоптерећен метод Metod(string s).
2) Програм има грешку, јер се b.Метод(5) не може позвати пошто је метод Metod(int i) заклоњен у класи B.
3) Програм има грешку због b.i, јер је поље i неприступачно из класе B.
4) Програм има грешку, јер је метод Metod(int i) надјачан (предефинисан) са различитим потписом у класи B.

РЕШЕЊЕ:
Две методе могу да имају исти назив уколико имају различит тип или број параметара. Овде је случај да једна метода са називом Metod има за тип параметара број (int), а друга за тип параметара има текст (string).

11. Заокружити наредбе којима се може допунити дефиниција конструктора:

```
public class Point {
 private double x, y;
 public Point() { x = 0; y = 0; }
 public void set(double xx, double yy) { x = xx; y = yy; }
 public Point(double x, double y) { _____________; } }
```

1) x=this.x; y=this.y;
2) x=x; y=y;
3) set(x,y);
4) set(this.x, this.y);
5) this.x=x; this.y=y;

РЕШЕЊЕ:
Потребно је разумети коју променљиву представља this.x (ону која припада класи и коју желимо да променимо) и самим тим препознати изразе који одговарају траженом циљу.

12. Одредити који од понуђених одговора су исправни.

1) public void Print(params string[] niska, params double[] broj)
2) public void Print(int n, params double[] broj)
3) public void Print(params double[] broj, string niska)
4) public void params Print(double d1, double d2)
5) public void Print(params double[] broj)

РЕШЕЊЕ:
Код задатка са params-ом важно је запамтити да он мора да буде на последњем месту уколико се налази у параметрима методе (не сме да се појављује више пута и не сме да постоји ништа после њега у загради).

13. Анализирати код и на предвиђене линије уписати шта метод Uvecaj() враћа при позиву из наведених објеката:

```
public class Racun {
 public virtual int Uvecaj() { return 10; }}
public class Dinarski: Racun {
 public override int Uvecaj() { return 20 *
base.Uvecaj(); }}
public class Devizni : Racun {
 public override int Uvecaj() { return 50 +
base.Uvecaj(); }
```

```
Racun r = new Racun();
Racun rDin = new Dinarski();
Racun rDev = new Devizni();
```

r.Uvecaj(); ____10____
rDin.Uvecaj(); ____200___
rDev.Uvecaj(); ____60____

РЕШЕЊЕ:
Потребно је обратити пажњу на то која је базна класа (коју класу наслеђује она коју гледамо) и на основу тога израчунати вредност израза. Исто важи и за задатак испод.

14. Анализирати код и на предвиђене линије уписати шта метод Uvecaj() враћа при позиву из наведених објеката:

```
public class Racun {
 public virtual int Uvecaj() { return 10; }}
public class Dinarski: Racun {
 public override int Uvecaj() { return 20 *
base.Uvecaj(); }}
public class Devizni : Dinarski {
 public override int Uvecaj() { return 50 +
base.Uvecaj(); }
```

```
Racun r = new Racun();
Racun rDin = new Dinarski();
Racun rDev = new Devizni();
```

r.Uvecaj(); ____10____
rDin.Uvecaj(); ___200____
rDev.Uvecaj(); ___250____

15. На предвиђене линије уписати шта метод Metod() враћа при позиву из наведених објеката:

```
public class KlasaA { public virtual int Metod() { return 10; } }
public class KlasaB : KlasaA { public override int Metod() { return 20; } }
public class KlasaC : KlasaB { public new int Metod() { return 30; } }
```

KlasaA a = new KlasaA(); __10__ KlasaA bb = new KlasaB(); __20__ KlasaB cc = new KlasaC(); __20__

KlasaB b = new KlasaB(); __20__ KlasaC c = new KlasaC(); __30__ KlasaA ccc = new KlasaC(); __20__

16. Проценити ефекат извршења наведених позива и на предвиђене линије уписати шта ће се видети на стандардном излазу извршењем позваних метода:

```
public class Roditelj {
 public virtual void Poruka1(){
Console.WriteLine("R1"); }
 public void Poruka2(){ Console.WriteLine("R2");
}}
```

```
public class Dete : Roditelj {
 public override void Poruka1(){
Console.WriteLine("D1"); }
 public new void Poruka2(){
Console.WriteLine("D2"); }}
```

```
Dete x = new Dete();  Roditelj y = new Dete();
```

x.Poruka1(); __D1___ x.Poruka2(); __D2___ y.Poruka1(); __D1___ y.Poruka2(); __R2___

17. На линију испред описа уписати редни број под којим је наведен одговарајући тип класе:

1. sealed	_3_ Класа која се простире у више фајлова
2. abstract	_4_ Класа садржи само декларације метода, али не и дефиницију (тело) методе
3. partial	_2_ Класа која се не може инстанцирати
4. interface	_1_ Класа из које се не може наслеђивати

18. Шта од наведеног представља скуп специјализованих програма са функцијом веб сервера:

1) Microsoft NT Server
2) Microsoft SQL Server
3) Apache Web Server
4) Microsoft Internet Information Services

19. Написати линију кода којом се укључује екстерна CSS датотека style.css у оквиру заглавља веб странице index.html (датотеке style.css и index.html се налазе у истом директоријуму):

<link rel="stylesheet" href="style.css">

20. Повежите дате тагове и атрибуте са њиховим дефиницијама:

1) METHOD	_3_ Дефинише где проследити податке са форме после предаје (submit) форме
2) INPUT	_1_ Одређује начин на који се подаци са форме шаљу на дефинисано одредиште
3) ACTION	_4_ Основни таг формулара са којим се креира формулар за унос података од стране корисника
4) FORM	_2_ Дефинише поље за унос податка унутар HTML форме

21. Заокружити исказ који описује улогу Domain Name Server-а:

1) хостовање веб сајта
2) превођење имена домена у IP адресу
3) вршење функције главног чвора у локалној рачунарској мрежи
4) приказ динамичких веб страница

22. Заокружити број испред назива Microsoft-ове платформе за развој веб апликација:

1) HTML 2) JSP 3) PHP 4) ASP.Net

23. Заокружити исказ који дефинише улогу Proxy сервера:

1) Омогућава пренос датотека са удаљеног рачунара и ка удаљеном рачунару
2) Побољшава перформансе конекције, филтрира захтеве и прослеђује их на прави сервер
3) Пружа хостинг различитим медијским садржајима (Аудио, Видео)
4) Хостује веб стране

24. Током извршавања апликације написане у JavaScript-у:

1) Типови променљивих се могу мењати током извршавања програма
2) Није могуће мењати типове променљивих у току извршавања апликације
3) Типови променљивих се обавезно мењају током извршавања апликације у одговарајући веб тип променљиве
4) JavaScript не подржава типове промељивих

25. Заокружити функције које омогућавају конверзије стринга у број у JavaScript-у:

1) Није могуће мењати типове променљивих у току извршавања апликације
2) Функција tryParseINT
3) Функција parseINT
4) Функција parseFLOAT
5) Функција parseDOUBLE
6) Функција EVAL

26. Заокружити елементе који су укључени у .NET Framework:

1) класе корисника
2) скрипт језик који се извршава на клијент страни
3) класе корисничког интерфејса
4) веб сервер и примере базе података
5) класе за манипулацију XML докумената
6) класе за приступ подацима и базама

27. Повезати објекат са одговарајућим описом уносом редног броја којим је објекат нумерисан на означену линију:

1) history
2) location
3) window
4) document

_1_ Садржи URL адресе које су претходно биле посећене, као и URL адресе које су посећене након посете тренутној страници

_4_ Представља тренутно учитани документ и служи као приступна тачка садржају странице

_2_ Садржи информације о URL адреси документа који је тренутно приказан у посматраном прозору

_3_ Објекат највишег нивоа који садржи све друге објекте и представља прозор претраживача

28. Повежите метод и одговарајућу акцију корисника:

1) focus
2) submit
3) load
4) blur

_4_ Изађе из фокуса елемента форме
_3_ Учита страницу у прегледач
_1_ Уђе у фокус неког елемента форме
_2_ Изврши слање форме

29. Повезати категорију компоненти са одговарајућом групом:

1) HTML контроле
2) Контроле за податке
3) Серверске контроле
4) Системске контроле

_3_ TextBox, Label, Button, ListBox, DataGrid
_4_ FileSystemWatcher, EventLog, MessageQueue
_2_ SqlConnection, SqlCommand
_1_ Text Area, Table, Image, Submit Button

30. Повежите називе модула са функцијама које обављају:

1) HTTP модули
2) Безбедносни модули
3) Модули за евиденцију и дијагностику
4) Модули садржаја

_3_ Модули за праћење и извештавање о догађајима насталим током процесирања захтева...
_2_ Модули за ауторизацију УРЛ адреса, филтрирање захтева...
_4_ Модули за обраду захтева за статичке фајлове, враћање подразумевне странице када клијент не наведе ресурс у захтеву, излиставање садржаја директоријума...
_1_ Модули за одговорарање на информације, враћање HTTP грешака, преусмеравање захтева...

31. Одредити оператор који би требало користити у WHERE клаузули SELECT наредбе да би били приказани само они ученици чије презиме почиње словом А:

1) IN
2) IS LIKE
3) BETWEEN
4) LIKE
5) BEGINS WITH

32. Одредити оператор који треба употребити у WHERE клаузули SELECT наредбе да би били приказани сви подаци код којих није позната и није унета вредност у колону:

1) =NULL
2) IS NULL
3) ==NULL
4) ISNULL
5) LIKE NULL

33. Заокружити оператор који треба користити у датом упиту да би се избегло вишеструко коришћење оператора OR:

SELECT * FROM ucenici WHERE odeljenje=4 OR odeljenje=7 OR odeljenje=10

1) IN 2) BETWEEN 3) AND 4) LIKE

34. Заокруживањем редног броја обележити клаузулу коју је потребно користи уколико листа иза резервисане речи SELECT садржи агрегатну функцију и једну или више колона које нису део агрегатне функције:

1) GROUP BY 2) ORDER BY 3) HAVING 4) JOIN

35. Одредити шта ће се десити након извршења упита: ALTER TABLE PROJEKAT ADD RokKraj date

1) у табелу PROJEKAT додаје се колона RokKraj
2) у табелу PROJEKAT додаје се ограничење RokKraj
3) у табели PROJEKAT биће преименована колона
4) у базу података додаје се табела PROJEKAT са само једном колоном у табели PROJEKAT промениће се тип података у колони RokKraj

36. Одредити шта је резултат упита. Заокружити број испред траженог одговора:

SELECT Ime, Prezime, DATEDIFF(year, DatZap, GETDATE()) AS God FROM Radnik

1) Табела са подацима о именима и презименима радника
2) Табела са подацима о именима, презименима и броју година које су протекле од датума запослења радника до тренутног датума
3) Табела са подацима о именима, презименима и датумима запослења радника
4) Табела са подацима о именима, презименима и броју година које су протекле од датума запослења радника до краја века

РЕШЕЊЕ:
Функција DATEDIFF ће вратити разлику у опсегу који се први наведе, у овом случају године (year) од првог унесеног до другог унесеног датума. DatZap представља датум запослења, а функција GETDATE() враћа тренутни датум као вредност.

37. Одредити резултат извршавања датог упита. Заокружити број испред траженог одговора:

```
SELECT Imeod, AVG(Plata) AS ProsekPlata FROM Radnik, Odeljenje
WHERE Odeljenje.Brod=Radnik.Brod GROUP BY Imeod
HAVING AVG(Plata)>1000
```

1) Приказују називи одељења и висина просечне плате у њима само за одељења у којима је просечна плата већа од 1000
2) Приказују називи одељења и висина просечне плате у њима, при чему се код одређивања просека узимају у обзир само плате веће од 1000
3) Упит се не извршава зато што груписање мора да се изврши не само по називу одељења, него и по шифри одељења (BrOd)
4) Групишу по одељењима радници са платом већом од просечне плате

38. Одредити шта се види као резултат датог упита:

```
SELECT Odeljenje.Imeod, Radnik.Ime+' '+ Radnik.Prezme as PunoIme
FROM Odeljenje LEFT JOIN Radnik ON Radnik.Brod = Odeljenje.Brod
```

1) Називи свих одељења – и оних у којима има радника и оних где нико није распоређен - са бројем радника у сваком одељењу
2) За сваког распоређеног радника приказује се по један ред са називом одељења и пуним именом радника, док се за раднике који нису распоређени приказује само пуно име радника
3) Приказује се по један ред за сваког распоређеног радника са називом одељења и пуним именом радника. За раднике који нису распоређени, као и за одељења у која нико није распоређен, не формирају се редови у резултујућој табели
4) За свако одељење се приказује онолико редова колико радника ради у том одељењу, док се за одељења у којима нико не ради приказује по један ред са називом одељења

РЕШЕЊЕ:
LEFT JOIN значи да ће се приказати сви подаци из леве табеле и они подаци из десне који су повезани са левом табелом. У овом случају биће исписана сва одељења, а уколико неко одељење нема радника у себи поље за име и презиме радника ће остати празно.

39. Одредити приказ који је резултат датог упита. Заокружити број испред траженог одговора:

```
SELECT Odeljenje.Brod, Odeljenje.Imeod, COUNT(*)
FROM Radnik INNER JOIN Odeljenje ON Radnik.Brod = Odeljenje.Brod GROUP BY Odeljenje.Brod, Odeljenje.Imeod
```

1) Бројева и назива свих одељења
2) Бројева и назива свих одељења са бројем радника у њима
3) Бројева и назива одељења у којима има радника са бројем радника у њима
4) Бројева и назива одељења у којима нема радника

РЕШЕЊЕ:
INNER JOIN значи да ће се приказати само подаци који се преклапају у обе табеле, односно у овом случају само одељења која имају раднике, а функција COUNT служи да преброји број радника у самом одељењу.

40. Одредити које ће вредности бити приказане након извршења упита:

```
SELECT MIN(Datum_Zaposlenja), Odsek_Id
FROM Zaposleni
GROUP BY Odsek_Id
```

1) Најранији датум запослења за сваки одсек предузећа.
2) Најранији датум запослења у целом предузећу.
3) Датум запослења последњег запосленог радника у целом предузећу.
4) Датум запослења последњег запосленог радника за сваки одсек.
5) Датум запослења најстаријег запосленог радника у сваком одсеку предузећа.

41. Потребно је креирати извештај који приказује имена свих производа чија је цена већа од просечне цене свих производа. Заокружити број испред упита који одговара постављеном задатку:

1) SELECT naziv
FROM proizvod
WHERE cena > AVG(cena)

2) SELECT naziv
FROM (SELECT AVG(cena) FROM proizvod)
WHERE cena > AVG(cena)

3) SELECT naziv
FROM proizvod
GROUP BY naziv
HAVING cena > AVG(cena)

4) SELECT naziv
FROM proizvod
WHERE cena > (SELECT AVG(cena) FROM proizvod)

РЕШЕЊЕ:
Агрегатне функције попут AVG, MAX, MIN се не могу налазити унутар WHERE клаузуле што одмах искључује први и други пример као неисправне. Трећи пример је неисправан јер HAVING клаузула неће давати добре резултате уколико се агрегатна функција не налази и у SELECT клаузули.

42. Наредба се неће извршити. Заокружити број испред разлога услед кога се наредба неће извршити:

```
SELECT prezime, ime, email
FROM ucenik
ORDER BY prezime
WHERE prosek>=4.50
```

1) Услов треба написати у HAVING клаузули
2) Наредба се неће извршити једино ако нема ни једног одличног ученика.
3) Потребно је променити редослед клаузула.
4) Потребно је назначити редослед сортирања (asc, desc).

43. Изабрати упит који даје извештај о минималној цени артикла у свакој категорији:

1)
```
SELECT kategorija, MIN(cena)
FROM artikli
WHERE MIN(cena)>@granica
GROUP BY cena
```

2)
```
SELECT kategorija, MIN(cena)
FROM artikli
GROUP BY kategorija
HAVING MIN(cena)>@granica
```

3)
```
SELECT kategorija, MIN(cena)
FROM artikli
GROUP BY MIN(cena), kategorija
HAVING MIN(cena)>@granica
```

4)
```
SELECT kategorija, MIN(cena)
FROM artikli
WHERE MIN(cena)>@granica
```

РЕШЕЊЕ:
Први и четврти пример су неисправни јер садрже агрегатну функцију унутар WHERE клаузуле док је трећи пример неисправан јер групише и по минималној цени поред категорије уместо само по категорији.

44. Заокружити број испред тачног исказа:

```
CREATE VIEW Pregled_Proseka AS
SELECT UcenikID, Ime, Prezime, AVG(Ocena) AS Prosek FROM Testovi
WHERE OdeljenjeID IN (1, 2, 3, 4)
GROUP BY UcenikID, Ime, Prezime
```

1) Подаци у табели Testovi се могу модификовати коришћењем погледа Pregled_Proseka
2) Коришћењем датог погледа, подаци из табеле Testovi се могу само прегледавати, али не и додавати или мењати
3) Овако дат упит изазива грешку при извршењу
4) Коришћењем датог погледа, подаци се могу само у додавати у табелу Testovi, али не и мењати

45. Обележити команде које се сматрају командама ажурирања података у бази података.

1) Организовање податка
2) Додавање нових података
3) Додела права приступа подацима
4) Враћање оштећених података у коректно стање
5) Измена постојећих података
6) Брисање старих података
7) Измена структуре постојећих табела у бази

46. За упите са специфицираним редоследом приказа врста у резултујућој табели користи се клаузула ORDER BY после које се наводи назив колоне:

1) и службена реч DESC за растући редослед
2) и службена реч ASC за опадајући редослед
3) и службена реч DESC за опадајући редослед
4) и службена реч ASC за растући редослед
5) службена реч се може изоставити при чему се добија растући поредак
6) службена реч се може изоставити при чему се добија опадајући поредак

47. Заокружити бројеве испред наредби које служе за креирање, брисање и измену структуре релационе базе и објеката који чине релациону базу:

1) ALTER TABLE
2) ADD CONSTRAINT
5) DELETE TABLE
6) UPDATE
3) CREATE TABLE
4) DROP TABLE
7) INSERT
8) ADD COLUMN

48. Дати су искази који се односе на спољашњи кључ табеле (foreign key). Заокружити бројеве испред тачних исказа:

1) Вредност у пољу спољашњег кључа не сме бити NULL
2) Више редова у табели може садржати исту вредност у пољу спољашњег кључа и тиме показивати на исти ред у референцираној табели
3) Вредност спољашњег кључа мора бити јединствена (unique) у колони над којом је постављено ограничење спољашњег кључа
4) Вредност у пољу спољашњег кључа мора бити или NULL или једнака некој од вредности из колоне на коју спољашњи кључ референцира
5) Колона спољашњег кључа не мора садржати исти тип података као колона на коју спољашњи кључ референцира

РЕШЕЊЕ:
Спољашњи кључ (foreign key) се односи на оно што једна табела узима из друге како би била повезана са њом. Самим тим вредност тог поља мора бити или празна или једнака некој од вредности у пољу табеле с којом се повезује.
Пример: Уколико би сви ученици вашег разреда били повезани са табелом школе ID школе би био страни кључ и био би једнак сваком од вас. Након завршетка школе то поље може остати празно.

49. Одредити који од понуђених исказа су тачни:

```
UPDATE Radnik SET Radnik.SifraOD = 30 WHERE Radnik.SifraOD=@br or @br IS NULL
```

1) Сви радници који раде у одељењу са шифром једнакој вредности која је пренета кроз параметар @br, биће прераспоређени у одељење чија је шифра 30
2) Упит распоређује све нераспоређене раднике у одељење са шифром 30
3) Уколико се параметру @br не пренесе вредност, радници који су до тог момента били нераспоређени, биће распоређени у одељење са шифром 30
4) Уколико се параметру @br не пренесе вредност, СВИ радници ће бити распоређени у одељење чији је brod 30

РЕШЕЊЕ:
Унутар табеле Radnik мењају се шифре одељења (SifraOD) за све раднике чија је тренутна шифра одељења једнака вредности параметра @br или свима уколико параметру @br није додељена никаква вредност.

50. Дати су искази који описују ефекат извршења упита. Заокружити бројеве испред ТАЧНИХ исказа:

SELECT Zaposleni_Id, Ime, Prezime FROM Zaposleni WHERE Plata=(SELECT MAX(Plata) FROM Zaposleni GROUP BY Odsek_Id)

1) Упит се не извршава зато што у подупиту није дозвољено коришћење групних функција.
2) Упит се извршава без грешке и из сваког одељења бира и приказује податке о раднику који има највећу плату у том одељењу.
3) Упит се не извршава јер подупит враћа више од једне врсте, а коришћен је оператор за поређење са једном вредношћу.
4) Уколико би се изоставила GROUP BY клаузула, упит би се извршавао без грешке и приказао би једног или више радника са платом једнакој највећој плати (без обзира на одељење).
5) Како подупит садржи груписање, да би се цео упит извршио без грешке, потребно је услов са подупитом написати у HAVING уместо у WHERE клаузули.

РЕШЕЊЕ:
Уколико користимо оператор = вредност са леве стране израза можемо поредити само са једном вредношћу, што са клаузулом GROUP BY неће бити могуће јер ће она вратити максималне вредности плате за сваки одсек посебно. Ако изоставимо поменуту клаузулу вратиће нам се само једна вредност највеће плате и упит ће бити исправан.

51. Заокружити бројеве под којима су наведене SQL наредбе у којима се могу користити аритметичке операције:

1) SELECT
2) FROM
3) WHERE
4) ORDER BY

52. Одредити које две вредности може вратити ова наредба:

```
SELECT cena
FROM proizvod
WHERE cena IN (101,125,150,350)
AND (cena BETWEEN 125 AND 140 OR cena >150)
```

1) 101
2) 150
3) 125
4) 110
5) 350

53. Заокружити бројеве испред команди које се могу користити за ажурирање постојећих података у бази:

1) MERGE
2) DELETE
3) UPDATE
4) SELECT

54. Заокружити бројеве испред кључних речи које се не користе за обележавање ограничења (constraints) у језику SQL:

1) Foreign key
2) Unique
3) Distinct
4) Check
5) Convert
6) Union
7) Not Null
8) Except

РЕШЕЊЕ:
Foreign key је ограничење које се односи на страни кључ (детаљније у одговору на 48. питање).
Unique је ограничење које се односи на то да вредност тог поља у табели мора бити јединствено.
Distinct користимо приликом креирања упита и утиче само на приказ података (јединствених, без понављања) и није ограничење у самој табели.
Check је ограничење које користимо да спречимо унос вредности које нам не одговарају.
Convert мења вредност из једног у други тип података приликом креирања упита.
Union користимо како би спојили неке податке, табеле или вредности при креирању упита и приказу података.
Not Null је ограничење које спречава да поље над којим је написано буде празно.
Except није ограничење већ означава које податке желимо да изоставимо из приказа података.

55. Одредити тачан исказ о оператору ANY који се примењује са подупитом који враћа више вредности:

1) Оператор ANY може да се користи испред кључне речи DISTINCT.
2) Оператор ANY врши поређење са свим вредностима које враћа подупит и враћа TRUE ако било која од вредности подупита задовољава услов
3) Оператор ANY врши поређење са свим вредностима које враћа подупит и враћа TRUE ако све вредности подупита задовољавају услов
4) Оператору ANY мора да претходи оператор поређења (=, <>, >, >=, <, <=)
5) Оператору ANY може да претходи оператор LIKE или оператор IN.
6) Услов =ANY(скуп вредности) је еквивалентан услову IN (скуп вредности)

56. Написати на цртама испред логичких операција редне бројеве њихових приоритета:

1) највиши приоритет _3_ OR
2) средњи приоритет _1_ NOT
3) најнижи приоритет _2_ AND

57. Свакој групи градова придружити по један услов уносом редног броја коим је услов нумерисан на линију испред листе градова:

1. where Naziv like 'L__ %' _3_ SIJERA LEONE, SVETA LUCIJA
2. where Naziv like '__ %N%' _1_ LAS VEGAS, LOS ANGELES
3. where Naziv like '% L%' _4_ EL SALVADOR, EL RENO
4. where Naziv like '_L%' _2_ EL RENO, LA KORUNJA
5. where Naziv like '__ %A' _5_ LA VALETA, LA KORUNJA

РЕШЕЊЕ:
Приликом коришћења LIKE оператора поредимо вредност неког поља са изразом који креирамо.
_ представља један карактер
% представља низ карактера
Размак представља размак између речи/карактера

58. Повезати упите и њихова значења уписом броја упита на одговарајућу линију:

1) SELECT odeljenje.imeod, radnik.prezime
FROM odeljenje INNER JOIN radnik
ON radnik.brod = odeljenje.brod
2) SELECT odeljenje.imeod, radnik.prezime
FROM odeljenje LEFT JOIN radnik
ON radnik.brod = odeljenje.brod
3) SELECT odeljenje.imeod, radnik.prezime
FROM odeljenje RIGHT JOIN radnik
ON radnik.brod = odeljenje.brod

_3_ Приказује све раднике (и који јесу и који нису распоређени у одељења) и само она одељења у којима има радника

_1_ Приказује само одељења у којима има радника и само раднике распоређене у одељењима

_2_ Приказује сва одељења (и она у којима има и она у којима нема радника) и само оне раднике који су распоређени у одељења

59. Повезати упите и њихова значења уписом броја датог испред описа значења упита на одговарајућу линију:

_4_ SELECT odeljenje.imeod, radnik.prezime
FROM odeljenje LEFT JOIN radnik
ON radnik.brod = odeljenje.brod
WHERE radnik.brod IS NULL

_1_ SELECT odeljenje.imeod, radnik.prezime
FROM odeljenje RIGHT JOIN radnik
ON radnik.brod = odeljenje.brod
WHERE odeljenje.brod IS NULL

_3_ SELECT odeljenje.imeod, radnik.prezime
FROM odeljenje FULL JOIN radnik
ON radnik.brod = odeljenje.brod

1. Приказује само раднике који нису распоређени у одељења
2. Приказује све раднике (и који јесу и који нису распоређени у одељења) и само она одељења у којима има радника
3. Приказује сва одељења - и она у којима има и оне у којима нема радника и све раднике – и оне који су распоређени у одељења, као и оне који нису распоређени
4. Приказује само одељења у којима нема радника

60. На линију испред команде уписати број под којим је наведена категорија којој команда припада:

1) DDL - Data Definition Language
2) DML - Data Manipulation Language
3) DCL - Data Control Language
4) TCL - Transaction Control Language

_4_ COMMIT
_3_ GRANT
_2_ UPDATE
_1_ ALTER

_1_ DROP
_2_ DELETE