ПРОГРАМИРАЊЕ – ПРОГРАМСКИ ЈЕЗИК С

У следећим задацима заокружите број испред траженог одговора

1. Заокружити број испред кључне речи којом се при кодирању у програмском језику Ц, у наредби вишестуког гранања обележавају вредности за које се улази у поједине гране: **1**

   1. switch
   2. break
   3. return
   4. case

2. У програмском језику Ц декларисане су променљиве и дат је део кода програма: **1**

```
FILE *fp;
char str[150];
fp=fopen("primer.txt", "r");
fgets(str, 80, fp);
```

   Одредити шта је последица извршавања датог кода. Заокружити број испред очекиваног одговора:

   1. Учитава 80 карактера из датотеке и смешта у стринг str.
   2. Учитава максимално 150 карактера из датотеке и смешта у стринг str
   3. Учитава стринг из датотеке све док се не учита знак за нови ред или 80 карактера
   4. Учитава стринг из датотеке све док се не учита знак за нови ред или 150 карактера

3. У програму написаном у програмском језику Ц декларисана је променљива **pod** типа **int**. Употребом функције **fprintf(...)** уписати декларисан податак у стандардну излазну датотеку. **1**

   Заокружити број испред исправно написане наредбе:

   1. `fprintf(pod);`
   2. `fprintf("%d", pod);`
   3. `fprintf("%d", pod, stdin);`
   4. `fprintf(stdout, "%d", pod);`

4. У програмском језику Ц декларисани су структурни типови података ***Tacka3D*** (који дефинише тачку у простору) и ***Lopta*** (одређена центром и полупречником): **1**

```
typedef struct
{
  float x, y, z;
}Tacka3D;
```

```
typedef struct
{
  Tacka3D centar;
  float R;
}Lopta;
```

   Заокружити број испред исправно написане наредбе декларације и иницијализације променљиве ***x*** типа ***Lopta***, тако да јој центар буде у тачки O(2,2,2), а полупречник 10цм:

```
1. Lopta x={10, {2, 2, 2}};
2. Lopta x={2, 2, 2, 10};
3. Lopta x={2, 2, 2}, {10};
4. Lopta x={{2, 2, 2}, 10};
```

5. У програму написаном у програмском језику Ц декларисана је променљива fp која представља показивач на бинарну датотеку и променљива **podatak** у коју ће се уписати прочитани подаци из дефинисане бинарне датотеке.

Заокружити редни број испред наредбе која омогућава учитавање три бајта са тренутне позиције из бинарне датотеке:

1. `fread(podatak, 24, 1, fp);`
2. `fread(&podatak, 24, 0, fp);`
3. `fread(&podatak, 3, 1, fp);`
4. `fscanf(&podatak, 3, 1, fp);`
5. `fscanf(fp, 3, &podatak);`

1

6. У програму написаном у програмском језику Ц декларисана је променљива fp која представља показивач на бинарну датотеку и променљива **podatak** чија вредност ће се уписати у дефинисану бинарну датотеку.

Заокружити редни број испред наредбе која омогућава упис три бајта у бинарну датотеку:

1. `fprintf(&podatak, 3, 1, fp);`
2. `fprintf(fp, 3, &podatak);`
3. `fwrite(podatak, 24, 1, fp);`
4. `fwrite(&podatak, 24, 0, fp);`
5. `fwrite(&podatak, 3, 1, fp);`

1

7. У програмском језику Ц дата је наредба декларације, а затим и наредба форматираног излаза:

```
float x = 5.56;
printf(" x = %f\tx = %g\n", x, x);
```

Након извршења ових наредби на екрану ће се приказати вредности променљивих у задатом формату. Заокружити број испред тачног одговора:

1. x = 5.560000e+000 x= 0
2. x = 5.560000 x = 5.560000e+000
3. x = 5.560000 x = 5.56
4. x = 5.56 x = 5.560000e+000

2

8. У програмском језику Ц декларисана је целобројна променљива и додељена јој је вредност логичког израза:

```
int x;
x = 1==10>5;
```

Имајући у виду приоритет оператора, одредити вредност променљиве x после извршења ове наредбе. Заокружити број испред траженог одговора:

1. променљива добија вредност логичке неистине, тј. x = 0
2. променљива добија вредност логичке истине, тј. x = 1
3. вредност логичког израза се не може доделити целобројној променљивој
4. променљива добија вредност логичке истине, тј. било који број различит од 0

2

9. У програмском језику Ц декларисане су две целобројне променљиве. Променљива **a** добија вредност уносом са тастатуре. Затим се вредност логичког израза додељује променљивој **x**:

```
int x, a;
scanf("%d", &a);
x = 10!=5 || a<2;
```

Имајући у виду приоритет оператора, одредити вредност променљиве x после извршења ове наредбе. Заокружити број испред траженог одговора:

1. уколико се заградама не одредити редослед извршавања операција у овом изразу, долази до грешке, тј. „пуцања“ програма
2. без обзира на вредност која се унесе у променљиву **a**, вредност израза је увек „тачно“, тј. **x = 1**
3. без обзира на вредност која се унесе у променљиву **a**, вредност израза је увек „нетачно“, тј. **x = 0**
4. вредност израза зависи од променљиве **a** и не може се једнозначно одредити уколико није позната вредност уписана у променљиву **a**

2

10. Наредбом у Ц језику треба проверити да ли је број паран или непаран. Проценити која од датих наредби врши ову проверу и заокружити број испред тачно написане наредбе.

1. `(broj % 2 == 1) ? printf("PARAN!!") : printf("NEPARAN!!");`
2. `(broj % 2) ? printf("PARAN!!") : printf("NEPARAN!!");`
3. `(broj % 2 == 0) ? printf("PARAN!!") : printf("NEPARAN!!");`
4. `(broj & 1) ? printf("PARAN!!") : printf("NEPARAN!!");`
5. `(broj & 0x1 == 0) ? printf("PARAN!!") : printf("NEPARAN!!");`
6. `(broj & 1 == 1) ? printf("PARAN!!") : printf("NEPARAN!!");`

2

11. Код дат у тексту задатка треба реализовати помоћу једне if наредбе. Заокружити број испред понуђеног тачног одговора:

```
if(x>1)
{
      if(x<6)
            y=4;
}
```

1. `if(x>1 && x<6 ) y=4;`
2. `if(x>1 || x<6 ) y=4;`
3. `if(x<1 || x>6 ) y=4;`
4. `if(!(x<=1 || x>=6)) y=4;`

2

12. Дат је део кода на програмском језику Ц:

```
for(j=0; j<n; j++)
      if(a[j]>0)s+=a[j];
      else break;
```

Свака for петља може се написати коришћењем while и do-while наредбе. Заокружити број испред понуђеног кода који је еквивлентан коду датом у тексту задатка:

1. `j=0;`
   `while(j<n && a[j]>0) s+=a[j++];`
2. `j=0;`
   `while(j<n && a[j++]>0) s+=a[j];`
3. `j=0;`
   `while(j<n || a[j]>0) s+=a[j++];`
4. `j=0;`
   `while(j<n && a[j]<=0) s+=a[j++];`

2

13. Дата је декларација променљивих unsigned a, b и део кода у програмском језику Ц. Одредити шта се налази као резултат у променљивој x и у након извршења датог кода. Заокружити број испред траженог одговора:

```
unsigned  a, b, x, y, temp;
x=a*b;
while(b) temp=a%b, a=b, b=temp;
y=b;
x/=y;
```

1. x је производ а и б, а у је количник а са б
2. x је најмањи заједнички садржалац за а и б, а у највећи заједнички делилац за а и б
3. x је највећи заједнички делилац за а и б, а у најмањи заједнички садржалац за а и б
4. без обзира на вредности променљивих, долази до грешке у последњој наредби кода
5. долази до грешке јер петља понавља само прву наредбу услед изостанка витичастих заграда на телу петље

2

14. Дата је декларација променљивих pod, br и део кода у програмском језику Ц:

```
unsigned  pod, br;
pod=128;
br=0;
while(pod!=0){
      if(pod & 0x1) br++;
      pod>>=0x1;
}
```

Закључити шта представља вредност коју променљива br добије извршењем кода. Заокружити број испред траженог одговора:

1. Број јединица у бинарном запису броја pod
2. Број нула у бинарном запису броја pod
3. Број цифара у бинарном запису броја pod
4. Број цифара у хексадецималном запису броја pod

2

15. Дат је део кода на програмском језику Ц, који контролише унос целобројне променљиве n. Одредити вредности које променљива n може добити. Заокружити број испред траженог одговора:

```
do{
      printf("Unesite N:\nN = ");
      scanf("%d",&n);
      if(n & 1) printf("Greska.\n");
}while(n & 1);
```

1. Омогућава унос непарног природног броја
2. Омогућава унос само позитивног природног броја
3. Омогућава унос само негативног природног броја
4. Омогућава унос парног природног броја
5. Омогућава унос само непарног позитивног природног броја

2

16. Дата је декларација променљивих и део програмског кода:

```
int i, temp, n = 11;
int x[30]={ -3, -1, -2, -2, 1, 4, 3, 1, 5, -8, 5};
temp=x[0];
i=0;
while(i<n-1) x[i++]=x[i+1];
x[n-1]=temp;
```

Просудити на основу наредби које ће бити извршене у **while** циклусу како ће изгледати трансформисан низ **x од n елемената**. Заокружити број испред очекиваног одговора:

1. `x[ ] ={ 5, -3, -1, -2, -2, 1, 4, 3, 1, 5, -8 }`
2. `x[ ] ={ -1, -2, -2, 1, 4, 3, 1, 5, -8, 5, -3 }`
3. `x[ ] ={ -2,  0, -1, -1, 2, 5, 4, 2, 6, -7, 6 }`
4. `x[ ] ={ -1, -2, -2, 1, 4, 3, 1, 5, -8, 5 }`

2

17. У програмском језику Ц је дата декларација променљивих, а касније у коду извршен позив функције на следећи начин:

```
int k, i;
char lista[10][50], ime[50];
if( Formiraj(lista[i], ime, k) == NULL) { ... }
```

На основу позива, проценити каквог је облика прототип функције и заокружити број испред тачно написаног прототипа:

1. `void *Formiraj(char s1, char s2, int x);`
2. `char Formiraj(char *s1, char *s2, int x);`
3. `int *Formiraj(char s1[], char s2[], int x);`
4. `int Formiraj(char s1[], char s2[], int x);`
5. `char *Formiraj(char s1, char s2, int x);`

2

18. У програмском језику Ц је дата декларација променљивих, а касније у коду извршен позив функције на следећи начин:

```
int x, y, i, j;
float **mat, *vek, z;
mat[i] = Formiraj(x, 0.5);
```

На основу позива, проценити каквог је облика прототип функције и заокружити број испред тачно написаног прототипа:

1. `float Formiraj(int n, float m);`
2. `void *Formiraj(int n, int m);`
3. `float *Formiraj(float n, int m);`
4. `float *Formiraj(int n, float m);`
5. `float **Formiraj(int n, float m);`

2

19. Дат је код рекурзивне функције написан у програмском језику Ц:

```
void prikaz(int k, int n){
      printf("%d\t",k);
      if(k<n) prikaz(k+1,n);
      printf("%d\t",k);
}
```

Проценити шта ће се десити ако се функција позове наредбом: `prikaz(4,10);`

Заокружити број испред тачног одговора:

1. 4 5 6 7 8 9 10
2. 4 5 6 7 8 9 10 9 8 7 6 5 4
3. 4 5 6 7 8 9 10 10 9 8 7 6 5 4
4. 10 9 8 7 6 5 4

2

20. У програмском језику Ц дат је прототип функције **funkcija()** и декларисане су променљиве у функцији **main().** У понуђеним одговорима дати су позиви функције за декларисане променљиве.

```
void funkcija(int *x, int *y, int **p);
void main(){
      int a=5, b=7, c=15, *poc;
      poc = &c;
}
```

Заокружити редни број испред исправно записаног позива декларисане функције:

1. `funkcija(a, b, &poc);`
2. `funkcija(&a, &b, &poc);`
3. `funkcija(&a, &b, poc);`
4. `c = funkcija(&a, &b, &poc);`

2

21. У програмском језику С декларисани су структурни типови података ***Tacka*** (одређена координатама)***, Poligon*** (одређен бројем и координатама темена) и ***Piramida*** (одређена типом основе – троугао, четвороугао... и висином). Потом је декларисана и променљива типа ****Piramida***:

```
typedef struct
{
  float x, y;
}Tacka;
```

```
typedef struct
{
  int brojTemena;
  Tacka temena[10];
}Poligon;
```

```
typedef struct
{
  Poligon osnova;
  float visina;
}Piramida;
```

```
Piramida *p;
```

Заокружити број испред наредбе којом се број темена основе пирамиде на коју показује декларисани показивач ****p***, поставља на 6:

1. `p.osnova.brojTemena=6;`
2. `p.osnova->brojTemena=6;`
3. `p->osnova.brojTemena=6;`
4. `p->osnova[brojTemena]=6;`
5. `p->osnova->brojTemena=6;`

2

У следећим задацима заокружите бројеве испред тражених одговора

22. Заокружити бројеве испред **ТАЧНИХ** исказа који се односе на дефиницију while циклуса:

1. while циклус се извршава све док је услов логичка неистина (једнак нули),
2. while циклус се користи када се зна колико ће се пута циклус извршавати,
3. у while циклусу се увек прво проверава да ли је услов логичка истина, те ако јесте наредба се извршава
4. код while циклуса се може десити да се тело циклуса не изврши ниједном (на почетку услов није задовољен).

1

23. Наведени су искази који се односе на дефиницију **do-while** циклуса. Заокружити бројеве испред **ТАЧНИХ** исказа:

1. Користи се када се не зна колико ће се пута циклус понављати,
2. Прво се извршава тело циклуса, а затим израчунава вредност логичког израза. Ако се добије логичка неистина, циклус се поновно извршава.
3. Циклус се завршава када услов добија вредност логичке истине
4. Циклус се извршава барем једном.

1

24. Заокружити бројеве испред понуђених тврдњи које представљају тачне наставке изјаве која се односе на повратну вредност функције **fopen**:

При покушају да датотеку отворимо за читање, функција fopen...

1. ако датотека не постоји, изазива грешку која доводи до пуцања програма
2. ако датотека не постоји, креира празну датотеку, поставља се на њен почетак и враћа показивач на ту датотеку
3. враћа NULL показивач ако датотека не постоји
4. ако датотека постоји, враћа показивач на ту датотеку

1

25. Дате су наредбе декларације променљивих (са и без иницијализације вредности) написане на програмском језику Ц. Заокружити бројеве испред исправно написаних наредби декларације променљивих:

1. `int a=b=c=5;`
2. `int a=5, b=5, c=5;`
3. `char zn="a";`
4. `long a; b=5; c;`
5. `int a=0xf2;`
6. `char zn='\b';`

1,5

26. Декларисане су следеће променљиве:

```
float x, z;
const float y;
```

Заокружити бројеве испред НЕИСПРАВНО написаних наредби доделе вредности променљивама:

1. x %= y;
2. x =+ 5;
3. x += y + 5;
4. x =/ y + 5;
5. y = x + z;
6. x = z = y + 5;

1,5

27. Дата је наредба декларације **int a, b;**
Имајући у виду дату декларацију, заокружити бројеве испред НЕИСПРАВНО написаних наредби форматираног уноса података:

1. `scanf("%d%f", &a, &b);`
2. `scanf("%d%*d", &a, &b);`
3. `scanf("%d%d", &a, &b);`
4. `scanf("%d%d", a, b);`
5. `scanf("%d%*d", &a);`
6. `scanf("%5d%5d", &a, &b);`

1,5

28. Дате су наредбе декларације и иницијализације једнодомензионалног низа целих бројева у програмском језику Ц. Заокружити бројеве испред исправно написаних наредби декларације и иницијализације једнодимензионалног низа:

1. `int a[10]={1,2,3};`
2. `int a[5]={-3, -2, -1, 0, 1, 2, 3};`
3. `int a[]={10,20,30,40,50};`
4. `int[5] a={1, 2, 3, 4, 5};`
5. `int a={10,20,30,40,50};`
6. `int a[5]={'1', '2', '3', '4', '5'};`

1,5

29. Термин „адресна аритметика“ се односи на извођење аритметичких операција над показивачима. Анализирати дате исказе који дефинишу дозвољене аритметичке операције над показивачима. Заокружити бројеве испред тачних исказа:

1. Додела вредности једног показивача другом.
2. Додавање рационалног податка на вредност показивача и одузимање рационалног податка од вредности показивача.
3. Одузимање и упоређивање два показивача.
4. Идентификатор низа је показивач на почетак низа и може му се мењати вредност.
5. Упоређивање показивача са **NULL**.
6. Сабирањем два показивача, добија се нови показивач.

1,5

30. Наредбама програмског језика Ц дата је декларација једне симболичке константе и једне константне променљиве:

```
#define k 50 …
int m=100; …
```

Заокружити бројеве испред исправно написаних наредби декларације дводимензионалног низа целих бројева (матрице):

1. `int a [ k ][ k ];`
2. `int b [ k ][ m ];`
3. `int c [ k ][ 10 ];`
4. `int x [100 ][ 50];`
5. `int y [10, 10];`
6. `int z [ m ][ 10 ];`

1,5

31. У следећем задатку заокружити бројеве испред тражених одговора.
Дата је наредба у Ц језику, која температуру у Целзијусима ***tempc*** претвара у температуру у Фаренхајтима ***tempf***. Подаци tempc и tempf су реални бројеви обичне тачности. Проценити који изрази дају тачно решење.

1. temf = (9 / 5) * temc + 32;
2. temf = 9 / 5 * temc + 32;
3. temf = 9 * temc / 5 + 32;
4. temf = 32 + 9 * temc / 5;

2

32. Дата је if-else наредба:

```
if(a==3 || a==5)  p++;
else p--;
```

Заокружити бројеве испред понуђених switch наредби које су еквивалентне датој if-else наредби:

1.
```
switch(a){
            case 3: p++; break;
            case 5: p++; break;
            default: p--;
}
```
2.
```
switch(a){
            case 3: case 5: p++; break;
            p--;
}
```
3.
```
switch(a){
            case 3: case 5: p++; break;
            default: p--;
}
```
4.
```
switch(a){
            case 3: case 5: p++;
            default: p--;
}
```

2

33. У програму на програмском језику Ц извршена је следећа декларација, а касније и резервација меморијског простора за низ реалних бројева обичне талности дужине n:

```
float *B;
int n;
B=(float*)calloc(n,sizeof(float));
```

Заокружити бројеве испред исправно написаних наредби за **ПРИКАЗ** i-тог елемента низа B:

1. `printf("%f", B[i]);`
2. `printf("%f", &B[i]);`
3. `printf("%f", B+i);`
4. `printf("%p", *(B+i));`
5. `printf("%f", *(B+i));`

2

34. У програму на програмском језику Ц извршена је следећа декларација, а касније и резервација меморијског простора за низ реалних бројева обичне талности дужине n:

```
float *B;
int n;
B=(float*)calloc(n,sizeof(float));
```

Заокружити бројеве испред исправно написаних наредби за **УНОС** i-тог елемента низа B:

1. `scanf("%f", B[i]);`
2. `scanf("%f", B+i);`
3. `scanf("%p", B+i);`
4. `scanf("%f", &B[i]);`
5. `scanf("%f", *(B+i));`

2

35. Дати су прототипови функција написани у програмском језику Ц. Заокружити бројеве испред исправно написаних прототипова функција:

1. `float* pp1(int a, int b, int c);`
2. `int pp2(int a[][10], int n);`
3. `int pp3(int a[], n; float b);`
4. `void pp4(int *a, int n);`
5. `int pp5(int a[][], int n);`
6. `int pp6(int a[], int n);`
7. `int pp7(int a, b, c);`
8. `float[ ] pp8(float a[ ], int n);`

2

36. Заокружити бројеве испред понуђених тврдњи које представљају тачне наставке изјаве која се односе на повратну вредност функције **fopen**:

При покушају да датотеку отворимо за <u>писање</u>, функција fopen...

1. ако датотека не постоји, креира празну датотеку, поставља се на њен почетак и враћа показивач на ту датотеку
2. враћа NULL показивач ако датотека не постоји
3. ако датотека постоји, излази упозорење да ће њен садржај бити уништен при отварању
4. ако датотека не постоји, изазива грешку која доводи до пуцања програма
5. ако датотека постоји, уништава њен садржај без упозорења

2

37. У програмском језику Ц је декларисана низовна променљива:

```
int niz[10];
```

Заокружити бројеве испред исправно написаних наредби читања низа целих бројева дужине 10 из бинарног фајла на који показује показивач ***in**:

1. `fread(niz, sizeof (int), 10, in);`
2. `fread(&niz, sizeof (int), 10, in);`
3. `fread(&niz, sizeof niz, 1, in);`
4. `fread(niz, sizeof niz, 1, in);`
5. `fread(niz, sizeof (niz), 1, *in);`
6. `fread(niz, sizeof (int)*10, in);`

2

38. У програмском језику C декларисан је структурни тип података ***Ucenik***, а затим и променљива типа ***Ucenik***:

```
typedef struct
{
    char ime[50];
    int razred;
    int ocene[10];
}Ucenik; …
int i; Ucenik x;
```

Заокруживањем обележити исправне начине приступа пољима структурне променљиве ***x***:

1. `x.ocene[i]`
2. `*x.razred`
3. `x->ime`
4. `x[i].ocene`
5. `x.ime`

2

39. У програмском језику Ц декларисан је структурни тип података ***Putovanje***, а затим и променљива типа ****Putovanje***:

```
typedef struct
{
    char start[50], cilj[50];
    int kilometraza;
}Putovanje; …
Putovanje *p;
```

Заокруживањем обележити исправне начине приступа пољима структурне променљиве:

1. `*p->kilometraza`
2. `(*p).kilometraza`
3. `&p->kilometraza`
4. `p->start`
5. `*(p).start`

2

40. Дат је прототип функције написан у програмском језику Ц:

```
void Saberi(int n, int *a, int *b);
```

У main функцији дате су следеће декларације променљивих:

```
int x[50][50], y[50], m, j, i;
```

Заокружити бројеве испред исправно написаних позива декларисане функције:

1. `Saberi(m, y[i], y[i+1]);`
2. `Saberi(y[i], x[i], x[i+1]);`
3. `Saberi(m, y, x[i][j]);`
4. `Saberi(y, x[i], x[i+1]);`
5. `Saberi(10, y, x[0]);`
6. `Saberi(x[i][j], x[i], x[j]);`

3

41. Дат је прототип функције написан у програмском језику Ц:

```
void Umetni(char *a, char k);
```

У main функцији дате су следеће декларације променљивих:

```
char s1[20], *s2, s3;
```

Заокружити бројеве испред исправно написаних позива декларисане функције:

1. `Umetni(s2, s1[i]);`
2. `Umetni(s2, s1);`
3. `Umetni(s2, 'A');`
4. `Umetni(s1, s3);`
5. `Umetni(*s2, s3);`
6. `Umetni(s3, &s1);`

3

Допуните следеће реченице и табеле

42. Дата су наредба декларације, а затим и наредба форматираног уноса вредности у променљиве, написана на програмском језику Ц:

```
int x, y;
scanf("%3i%3i", &x, &y);
```

Следи тастатурни унос у облику: **12345 12345**
За сваку променљиву одредити и на одговарајућу линију уписати, коју ће вредност променљива имати по извршењу наредби:

1. променљива x добија вредност **x =** 123
2. променљива y добија вредност **y =** 45

2

43. Дати је декларација променљивих `int a=3, b=15;`

Израчунати вредност коју ће променљиве имати по извршењу следеће наредбе:

```
b %= ++ a;
```

a = 4

b = 3

2

44. Одредити вредности које ће променљиве x и y имати по извршењу следећег кода:

```
int x=10;
int y=20;
if(x>50)
        x-=10;
        y+=10;
```

Уписати добијене вредности на предвиђене линије:

x = 10 y = 30

2

45. У програмском језуку Ц декларисане су две целобројне променљиве:

```
int x=0, izbor;
```

За дате вредности промељиве ***izbor***, одреди вредност промељиве *x* по извршењу следеће наредбе вишестуког гранања и уписати их на предвиђене линије:

```
switch(izbor)
{
case 1: x += 1;
case 2: x += 2; break;
case 3: x += 3;
default: x = 100;
case 4: x += 4;
case 5: x += 5;
}
```

1. за izbor=3, x= 109
2. за izbor=10, x= 109
3. за izbor=4, x= 9
4. за izbor=2, x= 2

2

46. Наредбама програмског језика Ц декларисана је правоугаона матрица и три целобројне променљиве:

```
int mat[10][20];  int x, N, M;
```

где **N** представља број врста, а **M** број колона правоугаоне матрице **mat**.

Допунити изразима који недостају кôд петље која има задатак да дуплира све елементе **последње колоне** матрице:

```
for(x = 0; x < N ; x++)
    mat[ x ][ M-1 ] *= 2;
```

2

47. Дата су следеће декларације: `int p[200], i, n, k;`

А затим и део кода који треба да из низа **p** дужине **n,** сажимањем **ИЗБАЦИ** елеменат низа са позиције k, а затим ажурира нову дужину низа.

Имајући у виду дату иницијализацију петље, у предвиђена поља унеси одговарајуће елементе **преписивањем израза** из листе понуђених израза (подразумевати да су све потребне променљиве иницијализоване):

```
for(i=k; i < n - 1 ; i++ )
    p[i]  = p[i+1] ;
n--;
```

1. p [i + 1]
2. p [i - 1]
3. p [i]
4. p [k]
5. i ++
6. i - -
7. < n
8. < n-1

2

48. Наредбама програмског језика Ц декларисана је правоугаона матрица и три целобројне променљиве:

```
int mat[10][20];  int k, N, M;
```

где **N** представља број врста, а **M** број колона правоугаоне матрице **mat**.

Допунити изразима који недостају кôд петље која има задатак да дуплира све елементе **прве врсте** матрице:

```
for(k=0; k< M       ; k++)

    mat[    0     ][    k     ]*=2;
```

2

49. Дати су изрази формирани коришћењем математичких оператора. Водећи рачуна о типовима података, одредити вредности датих израза и уписати их линију у продужетку. Ако израз изазива грешку, уместо вредности, написати **error**:

1. 10 / 4. = 2.5
2. 10. / 5 = 2.0
3. -10 % 3 = -1
4. 10. % 5 = error
5. 10 % (-3) = 1
6. (100/3) % 6 = 3

3

50. Дате су следеће декларације: `int p[200], i, n, pom;`

А затим и део кода који треба да врши циклично померање елемената низа **p** дужине **n,** за једно место **удесно**. У коду недостају неки од елемената.

Имајући у виду дату иницијализацију петље, у предвиђена поља унеси одговарајуће елементе **преписивањем израза** из листе понуђених израза (подразумевати да су све потребне променљиве иницијализоване):

```
pom = p[n-1] ;
for(i=n-2; i >= 0 ; i--    )
      p[i+1]  = p[i]     ;
p[0]      = pom;
```

1. p [0]
2. p [n-1]
3. p [n]
4. p [i + 1]
5. p [i - 1]
6. p [i]
7. i ++
8. i - -
9. >=0
10. >0

3

51. Дата су следеће декларације: `int p[200], i, n, k, x;`

А затим и део кода који треба да у низ **p** дужине **n** УБАЦИ (инсертује) елемент x на позицију k, а затим ажурира нову дужину низа.

Имајући у виду дату иницијализацију петље, у предвиђена поља унеси одговарајуће елементе **преписивањем израза** из листе понуђених израза (подразумевати да су све потребне променљиве иницијализоване):

```
for(i=n; i ___> k___; ___i--___)
      ___p[i]___ = ___p[i-1]___;
___p[k]___ = x;
n++;
```

1. p [i + 1]
2. p [i - 1]
3. p [i]
4. p [k]
5. i ++
6. i - -
7. >= k
8. > k

3

52. У програмском језику Ц, дате су следеће декларације: `int A[50], i, n;`

Потребно је формирти вектор са следећим вредностима:

i=0	i=1	i=2	i=3	i=4	i=5	…	i=n-1
1	2	4	7	11	16	…	???

Допунити програмски код којим се формира овај вектор:

```
A[0]=1;
for(i = ___1___; i ___< n___; i++)
      ___A[i]___ = ___A[i-1] + i___;
```

3

53. У програмском језику Ц, декларисане су и иницијализоване променљиве:

```
int x=40, y=50, z=60, *p1, *p2;
```

Одреди које ће вредности имати променљиве x, y и z после извршења следећег кода и упиши на одговарајућу линију:

```
p1 = &x;
p2 = p1;
y = (*p2)+20;
z = *p2;
```

x = ___40___; y = ___60___; z = ___40___;

3

54. Дат је део кода написан на програмском језику Ц:

```
int a[7]={10,25,30,15,40,77,45}, *pa, x, y;
pa=a+4;
x=--(*pa)+5;
y=*(--pa)+5;
```

Анализирати код и одредити вредности променљивих **x** и **y**, као и показивача **pa,** по извршењу све три извршне наредбе датог кода:

x = 44

y = 20

pa = a + 3

3

55. Дат је део кода написан на програмском језику Ц:

```
int a[7]={81,12,35,97,40,52,17}, *pa, x, y;
pa=a+3;
x = *(pa-2)+1;
y = (*pa-2)+1;
```

Анализирати код и одредити вредности променљивих **x** и **y**, као и показивача **pa,** по извршењу све три извршне наредбе датог кода:

x = 13

y = 96

pa = a + 3

3

56. Дата је дефиниција функције:

```
void Transformisi(float *x, float *y, float z)
{
    z++;
    *x=*x+z;
    (*y)++;
}
```

У главном програму су декларисане променљиве и извршен је позив функције:

```
float a=10, b=10, c=10;

Transformisi(&a, &b, c);
```

Одредити које вредности имају променљиве a, b и c по изласку из функције и уписати их на одговарајућу линију:

a = 21

b = 11

c = 10

3

57. На програмском језику Ц декларисан је и иницијализован стринг и два показивача:

```
char s1[]="Short Message Service", *s2, *s3;
```

Одредити и на предвиђену линију уписати садржај означених стрингова по извршењу следећих наредби:

```
s2=strchr(s1,'M');
s3=strrchr(s2,'S');
strncpy(s1+1,s2,1);
strcpy(s1+2,s3);
```

s1 = SMService

s2 = ice

s3 = Service

3

У следећим задацима уредите и повежите појмове према захтеву

58. Са леве страни дати су допунски параметри у функцији printf , а са десне стране значење тих параметара у програмском језику Ц. На линију испред значења унети број којим је означен одговарајући допунски параметар:

1. **(#)** 3 — означава да ће се поравнавање вршити уз леву ивицу поља ширине **n** знакова, допунски знакови размака додају се иза, а не испред податка
2. **(0)** 4 — означава да се испред позитивног броја мора исписати знак плус
3. **(-)** 2 — нула код нумеричких података означава да ће се приликом равнања уз десну ивицу број допуњавати нулама, а не знаковима размак
4. **(+)** 1 — исписује се децимална тачка у конверзији рационалних бројева који немају разломљени део.

2

59. Декларисана је реална променљива `float w=123.456;`:

Са леве стране су дати различити прикази вредности променљиве добијени коришћењем наредби форматираног излаза које су приказане са десне стране. Поред сваке наредбе, на предвиђену линију уписати рени број приказа добијеног изршавањем те наредбе:

1. 123.456000 — 3 — `printf("%g", w);`
2. 1.234560e+002 — 1 — `printf("%f", w);`
3. 123.456 — 4 — `printf("%.2f", w);`
4. 123.46 — 2 — `printf("%e", w);`

2

60. Са леве стране дати су математички изрази, а са десне запис израза на програмском језику Ц. На линију испред записа израза, унети број којим је означен одговарајући израз:

1. $y = \frac{\sqrt{x}+10}{a+|b|}$
2. $y = \frac{\sqrt{x+10}}{a} + |b|$
3. $y = \frac{\sqrt{x+10}}{a+|b|}$
4. $y = \sqrt{x} + \frac{10}{a} + |b|$

3 y = sqrt(x+10) / (a+fabs(b))

4 y = sqrt(x)+10 / a+fabs(b)

2 y = sqrt(x+10) / a+fabs(b)

1 y = (sqrt(x)+10) / (a+fabs(b))

2

61. Дат је код на програмском језику Ц:

```
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;
}
```

Са десне стране су дате вредности променљиве с (скретница), а са леве стране резулат извршења кода за дату вредност скретнице. На линију испред вредности скретнице унети редни под којим је наведен одговарајући екрански приказ:

1. Krug Duz — 4 — 'b'
2. Pravougaonik Trougao Krug Duz — 6 — 'K'
3. Krug — 5 — 'A'
4. Trougao — 1 — 'c'
5. Pravougaonik Trougao
6. Duz

2

62. Са леве стране су набројани различити типови променљивих, а са десне су дате декларације променљивих у програмском језику Ц. На линију испред декларације унети редни број под којим је наведен одговарајући тип променљиве:

1. Једнодимензионални низ показивача на целе бројеве — 3 — `int *a;`
2. Вектор целих бројева — 2 — `int a[100];`
3. Показивач на цео број — 5 — `int a*[100];`
4. Цео број — 1 — `int *a[100];`
5. Грешка у декларацији

2

63. У програмском језику Ц, декларисан је показивач на цео број и функцијом **calloc** додељен му је простор за смештај низа од **n** целих бројева:

```
int *a, n;
scanf("%d", &n);
a=(int*)calloc(n, sizeof(int));
```

У левој колони дати су изрази, а у десној опис њиховог значења. На линију испред сваког од израза унеси број којим је означено одговарајуће објашњење:

3	`&a[0];`	1. вредност елемента на последњој позицији у низу
1	`*(a+n-1);`	2. адреса четвртог елемента у низу
6	`a+4;`	3. адреса почетног елемента низа
5	`*a;`	4. вредност елемента на предзадњој позицији низа
		5. вредност елемента на почетној позицији у низу
		6. адреса петог елемента у низу

2

64. Наредбама програмског језика Ц декларисано је дводиманзионално поље реалних бројева (матрица) и три целобројне променљиве:

```
float mat[10][10];  int i, j, n;
```

где променљива **n** представља димензију квадратне матрице **mat**.

Са леве стане су дате ознаке елемената матрице, а са десне њихово тумачење. На линију испред сваке ознаке унети редни број одговарајућег тумачења:

1	`mat[j][n-1]`	1. елеменат у j-тој врсти и последњој колони
3	`mat[j]`	2. и-та врста матрице
5	`mat[0][j]`	3. j-та врста матрице
2	`mat[i]`	4. j-та колона матрице
		5. елеменат у првој врсти и j-тој колони
		6. грешка у нотацији

2

65. Са леве стране су наведене наредбе позиционирања у датотеци, а са десне описи ефеката датих наредби. На линију поред наредбе уписати редни број под којим је наведен опис ефекта наредбе:

ftell(dat)	3	1. позиционирање на почетак датотеке
fseek(dat, 0, SEEK_END)	2	2. позиционирање на крај датотеке
fseek(dat, 0, SEEK_SET)	1	3. одређује позицију у датотеци у виду броја бајтова од почетка датотеке
rewind(dat)	1	4. ништа од понуђеног

2

66. Са леве стране дате су врсте конверзије, а са десне типови података који се користе у функцији за приказ података printf у програмском језику Ц. На линију испред типа података унеси број којим је означена одговарајућа конверзија:

1. %d	7	short
2. %i	1	signed int (u dekadnom obliku)
3. %s	4	long
4. %ld	8	unsigned
5. %f	2	signed int (dekadni, heksadekadni ili oktalni oblik)
6. %e		
7. %hd		
8. %u		

2,5

67. Са леве стране наведене су функције за читање и упис у текст датотеку, а са десне стране опис функције. На линију испред описа функције унети редни број под којим је наведена одговарајућа функција:

1. fscanf	5	учитавање карактера из датотеке
2. fgets	2	учитавање реда из датотеке
3. fputs	4	форматирани упис података у датотеку
4. fprintf	3	упис стринга у датотеку
5. fgetc	1	форматирано учитавање података из датотеке

2,5

68. Са леве стране су набројани неки од прелазних знакова тј. escape секвенце, а са десне стране дати су њихови описи. На линију испред описа упишите број под којим је наведена одговарајућа escape секвенца:

1. '\n'	3	враћање на почетак реда (carrage return)
2. '\t'	6	системски звучник (bell)
3. '\r'	1	прелаз у нови ред (new line)
4. '\b'	5	није escape секвенца
5. '\h'	2	хоризонтални табулатор (horizontal tab)
6. '\a'	4	враћање једну курсорску позицију назад (backspace)

3

69. Са десне стране наведене су неке од функција библиотеке ctype.h, а са леве су дати њихови описи. Испред назива сваке од наведених функција, уписати редни број под којим је дат одговарајући опис:

1. Да ли је с штампајући знак (укључујући и размак)?	6	isspace(c)
2. Да ли је с велико слово?	5	isdigit(c)
3. Да ли је с знак интерпункције?	7	isalpha(c)
4. Да ли је с управљачки знак?	2	isupper(c)
5. Да ли је с децимална цифра?	4	iscntrl(c)
6. Да ли је с знак бели знак?	1	isprint(c)
7. Да ли је с слово?		
8. Да ли је с хекса-децимална цифра?		

3

70. На програмском језику Ц декларисане су променљиве:

```
char s1[]="Iwnt2CmyM8sagain", *sn;
```

Са леве стране написани су изрази доделе вредности стрингу **sn**, а са десне стране понуђене су вредности стринга **sn**. На линију написати редни број под којим је наведена вредност стринга **sn** која се добија извшењем одговарајућег израза:

4	`sn=strrchr(s1,'a')-1;`	1. NULL
4	`sn=strchr(s1,'a')+1;`	2. “in”
6	`sn=strstr(s1,"my");`	3. “ain”
1	`sn=strstr(s1,"T2");`	4. “gain”
		5. “sagain”
		6. “myM8sagain”

4

ПРОГРАМИРАЊЕ – ПРОГРАМСКИ ЈЕЗИК C#

У следећим задацима заокружите број испред траженог одговора

71.	Дати су типови промењљивих у програмском језику C#. Одредити како се назива променљива која је дефинисана унутар неког метода. Заокружити број испред очекиваног одговора: 1. Глобална променљива 2. Статичка променљива 3. Блоковска променљива 4. Локална променљива	**1**
72.	Одредити какви могу бити чланови класе (поља и методе) у програмском језику C#. Заокружити број испред очекиваног одговора: 1. Локални и глобални 2. Процедурални и непроцедурални 3. Статички (класни) и нестатички (објектни) 4. Спољашњи и унутрашњи	**1**
73.	Одредити која поља су заједничка и јединствена за све креиране објекте неке класе дефинисане у објектно оријентисаном програмском језику C#. Заокружити број испред очекиваног одговора: 1. Јавна 2. Приватна 3. Објектна 4. Инстанцна 5. Статичка	**1**
74.	У програмском језику C# класа може да садржи статичка и не-статичка (инстанцна) поља. Дате су изјаве које се односе на статичка поља класе и међу њих је уметнута једна изјава која се односи на не-статичка (инстанцна) поља класе. Заокружити број испред изјаве која се односи на не-статичка поља класе: 1. Поље које се може се користити без конструисања иједног објекта те класе 2. Поље које има исту вредност за све креиране објекте неке класе 3. Поље чија се вредност може разликовати за сваки појединачни објекат неке класе 4. Поље које се може користити унутар статичких метода класе, као и унутар метода инстанце	**1**
75.	Заокружити број испред исказа који представља исправан наставак дате тврдње: При креирању објеката изведене класе... 1. извршава се само конструктор изведене класе 2. прво се извршава конструктор родитељске класе, али само ако је позван кључном речју ***base*** 3. обавезно се прво извршава конструктор изведене, а потом конструктор родитељске класе 4. обавезно се прво извршава конструктор родитељске, а потом конструктор изведене класе	**1**

76. У програмском језику C# користи се службена реч ***base***. Проценити који од наредних исказа који дефинишу дату службену реч **НИЈЕ** тачан.

Заокружити број испред очекиваног одговора:

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.

1

77. Дат је код програма у програмском језику C#:

```
namespace TestPrimer {
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine(fun(17));
        }
        public int fun(int n) { return n; }
        public void fun(int n){ Console.WriteLine(n); }
    }
}
```

Анализирати код и заокружити број испред очекиваног одговора:

1. Програм има грешку, јер се не може одредити коју верзију преоптерећеног метода fun(…) треба позвати.
2. Програм има грешку, јер је друга верзија преоптерећеног метода fun(…) дефинисана али се нигде не позива.
3. Програм се нормално извршава и приказује 17 једанпут.
4. Програм се нормално извршава и приказује 17 двапут.

2

78. Дат је код програма у програмском језику C# који формира и штампа елементе низа **a**. Анализирати дати код и проценити шта ће се догодити након његовог извршавања.

```
namespace TestPrimer {
    class Program {
        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] + " ");
        }
    }
}
```

Заокружити број испред очекиваног одговора:

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

2

79. У програмском језику C# дата је декларација низа:

int к;
int[ ] brojevi = {5, 12, 37, 7, 27, 33, 36};

На основу дате декларације одредити шта је резултат позива
k=Arrays.BinarySearch(brojevi, 37);

Заокружити број испред очекиваног одговора:

1. k=0, јер метод BinarySearch прво изврши сортирање низа у опадајућем редоследу, па онда тражи задату вредност
2. метод BinarySearch баца изузетак увек када је низ неуређен и програм „пуца“
3. k=2, јер се тражени елеменат налази на позицији 2
4. k добија неочекивану вредност јер низ мора бити сортиран у растућем поретку пре позива методе BinarySearch
5. k=6, јер метод BinarySearch прво изврши сортирање низа у растућем редоследу, па онда тражи задату вредност

2

80. У програмском језику C# дата је декларација једне стринг и једне целобројне променљиве, као и део кода:

```
string str = "Primer";
int broj = 66;
Console.WriteLine(str + broj + 65);
Console.WriteLine(broj + 65 + str);
```

Анализирати код и проценити шта ће се приказати на екрану након његовог извршења.
Заокружити број испред очекиваног одговора:

1. Primer6665
131Primer
2. Primer6665
6665Primer
3. Primer131
131Primer
4. PrimerBA
BAPrimer

2

81. Дат је код у програмском језику C#, који дефинише рекурзивни метод. Анализирати код и одредити резултат извршавања задатог метода.

```
public long fun(int n){
            return n * fun(n - 1);
}
```

Заокружити број испред очекиваног одговора:

1. Резултат позива fun(3) је 1.
2. Резултат позива fun(3) је 2.
3. Резултат позива fun(3) је 6.
4. Позив fun(3) изазива грешку јер производи бесконачан ланац позива истог метода fun(…).

2

82. Дат је код у програмском језику C#, који дефинише рекурзивни метод. Анализирати код и одредити резултат који ће се приказати на екрану.

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

Заокружити број испред очекиваног одговора:

1. Програм на екрану не приказује ништа
2. Програм на екрану приказује 1 2 3
3. Програм на екрану приказује 3 2 1.
4. Програм на екрану бесконачно приказује 1 1 1 1 1 ….
5. Програм на екрану бесконачно приказује 2 2 2 2 2 ….

2

83. У програмском језику C# дат је рекурзивни метод који проверава да ли је неки стринг палиндром. Да би код био комплетиран потребно је допунити трећи ред условом **if** наредбе.

```
1. public static bool palindrom(String s)
2. {
3.     if (s.Length <= 1) return true;  //bazni slučaj
4.     else if (____________________) return false;
5.     else return palindrom(s.Substring(1, s.Length - 2));
6. }
```

Заокружити број испред очекиваног одговора:

1. `s[0] != s[s.Length - 1]`
2. `s[0] != s[s.Length]`
3. `s[1] != s[s.Length - 1]`
4. `s[1] != s[s.Length]`

2

84. У програмском језику C# дат је рекурзивни метод који проверава да ли је неки string палиндром. Да би код био комплетиран потребно је допунити седми ред.

```
1. public static bool Palindrom(String s){
2.     return Palindrom(s, 0, s.Length - 1);
3. }
4. public static bool Palindrom(String s, int levi, int desni){
5.     if (desni <= levi) return true; // bazni slucaj
6.     else if (s[levi] != s[desni]) return false;
7.     else return _____________________;
8. }
```

Заокружити број испред очекиваног одговора:

1. `Palindrom(s)`
2. `Palindrom(s, levi, desni)`
3. `Palindrom(s, levi + 1, desni - 1)`
4. `Palindrom(s, levi + 1, desni)`
5. `Palindrom(s, levi, desni - 1)`

2

85. У програмском језику C# дат је рекурзивни метод за бинарно претраживање сортираног целобројног низа. Да би код био комплетиран потребно је допунити девети ред (означен линијом) помоћу понуђеног одговора.

```
1.  public static int TraziBroj(int[] niz, int broj) {
2.      return TraziBroj(niz, broj, 0, niz.Length - 1);
3.  }
4.  public static int TraziBroj(int[] niz,int broj,int levi,int desni) {
5.    if(levi > desni) return -1; // broj nije nadjen u nizu
6.    int sredina = (levi + desni) / 2;
7.    if(broj < niz[sredina]) return TraziBroj(niz, broj, levi,sredina-1);
8.    else if(broj > niz[sredina]) return ____________________________;
9.    else return sredina;
10. }
```

Заокружити број испред очекиваног одговора:

1. `TraziBroj(niz, broj, sredina + 1, levi)`
2. `TraziBroj(niz, broj, sredina - 1, levi)`
3. `TraziBroj(niz, broj, desni, sredina + 1)`
4. `TraziBroj(niz, broj, sredina + 1, desni)`

2

86. Дат је код програма у програмском језику C#. Анализирати дати код и проценити његову тачност. Заокружити број испред понуђеног тачног исказа:

```
namespace TestPrimer {
    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.

2

87. Дата је дефиниција класе у програмском језику C#. Проценити где у дефиницији класе (испред које методе) треба заменити знакове ***???*** службеном речју ***static***.

```
1.  public class Test {
2.      private int broj;
3.
4.      public ??? int kvadrant(int n) { return n * n; }
5.      public ??? int getBroj() { return broj; }
6.  }
```

Заокружити број испред тачне изјаве:

1. Метода ***kvadrant*** МОРА да буде статичка, док метода ***getBroj*** може и не мора.
2. Обе методе морају бити статичке.
3. Ни једна од дефинисаних метода није статичка.
4. Метода ***getBroj*** НЕ СМЕ да буде статичка, док метода ***kvadrant*** може и не мора.

2

88. Дата је дефиниција класе у програмском језику C# и састоји се од два конструктора, методе и поља x и y. У шестом реду написати конструктор копије објекта класе Point.

```
1.  public class Point {
2.      private double x, y;
3.      public Point() { x = 0; y = 0; }
4.      public void Set(double xx, double yy){ x=xx; y=yy; }
5.      public Point(Point p) {
6.          _____________________//Odgovor
7.      }
8.  }
```

2

Заокружити број испред очекиваног одговора:

1. `this(p.x, p.y);`
2. `this(p);`
3. `Set(p);`
4. `Set(p.x, p.y);`

89. Дат је код програма у програмском језику C# којим су дефинисане две класе: `class Program` која садржи `Main(string[] args)` методу и `class KlasaA`. Анализирати дати код и одредити да ли је код исправно написан. Понуђени одговори дају опис последица извршавања овог кода. Заокружити број испред тачног исказа.

```
class Program {
    public static void Main(string[] args){
        KlasaA a1 = new KlasaA ();
        KlasaA a2 = new KlasaA ();
        Console.WriteLine(a1.Equals(a2));
    }
}
class KlasaA {
    int x;
    public bool Equals(KlasaA a){
        return this.x == a.x;
    }
}
```

2

1. Програм има грешку, јер се изразом a1.Equals(a2) проверава једнакост објеката a1 и a2 различитог типа од Object.
2. Програм има грешку, јер се једнакост објеката a1 и a2 типа KlasaA проверава изразом a1 == a2.
3. Програм се извршава без грешке и приказује се true на екрану.
4. Програм се извршава без грешке и приказује се false на екрану.

90. Дат је код програма у програмском језику C# којим су дефинисане две класе: `class Program` која садржи `Main(string[] args)` методу и `class KlasaA`. Анализирати дати код и одредити да ли је код исправно написан. Понуђени одговори дају опис последица извршавања овог кода. Заокружити број испред тачног исказа.

```
class Program {
     public static void Main(string[] args){
            Object a1 = new KlasaA();
            Object a2 = new KlasaA();
            Console.WriteLine(a1.Equals(a2));
     }
}
class KlasaA {
     int x;
     public bool Equals(KlasaA a){
            return this.x == a.x;
     }
}
```

2

1. Програм има грешку, јер се изразом a1.Equals(a2) проверава једнакост објеката a1 и a2 различитог типа од Object.
2. Програм има грешку, јер се једнакост објеката a1 и a2 типа KlasaA проверава изразом a1 == a2.
3. Програм се извршава без грешке и приказује се true на екрану.
4. Програм се извршава без грешке и приказује се false на екрану.

91. Дат је код програма у програмском језику C# у ком су дефинисане три класе: class Program која садржи Main(string[] args) методу, class A и class B. Анализирати дати код и одредити да ли је код исправно написан.
Заокружити број испред исказа који даје информацију о тачности кода.

```
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);
  }
}
```

2

1. Програм има грешку, јер је метод **Metod(int i)** надјачан (предефинисан) са различитим потписом у класи B.
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 нема грешке, јер наслеђени метод класе A, **Metod(int i)** није надјачан у класи B, већ је дефинисан преоптерећен метод **Metod(string s).**

92. Дат је део кода који је написан на C# програмском језику. Анализирати код и одредити шта ће се приказати на излазу.

```
try
{
    int x = 0;
    int y = 5 / x;
}
catch (Exception e)
{
    Console.WriteLine("Exception");
}
catch (ArithmeticException ae)
{
    Console.WriteLine(" Arithmetic Exception");
}
Console.WriteLine("finished");
```

Заокружи број испред тачног одговора:

1. Приказује се текст: finished
2. Приказује се текст: Exception
3. Ништа. Дешава се грешка приликом компајлирања
4. Приказује се текст: Arithmetic Exception

2

93. Заокружити број испред исказа који представља исправан наставак дате реченице:

Ако try-catch наредба има више catch блокова у којима "хватамо" изузетак основне ***Exception*** класе, заједно са изузецима других класа изведених из класе ***Exceptions***...

1. онда се изузетак основне Exception класе може „хватати" у било ком catch блоку (редослед није битан, битно је да се наведу све могуће грешке)
2. онда се изузетак основне Exception класе мора „хватати" у последњем catch блоку
3. онда се изузетак основне Exception класе мора „хватати" у првом catch блоку
4. основна Exception класа се не комбинује у истој наредби са класама изведеним из ње јер их оснавна класа „маскира"

2

У следећим задацима заокружите бројеве испред тражених одговора

94. Дати су изкази који се односе на правила писања try-catch-finally блокова за руковање изузетима. Заокружити бројеве испред исказа који су тачни:

1. Блок try мора имати бар један catch или један finally блок
2. Блок try може имати више catch блокова
3. Ако блок try има више catch блокова, изузетак основне Exception класе мора се хватати у првом catch блоку
4. Ако блок try има више catch блокова, битан је редослед њиховог писања
5. Блок try мора имати бар један finally блок
6. Блок try не сме да има више catch блокова

1,5

95. Да би наслеђени метод могао да се **редефинише** и тиме измени његова функционалност у класама наследницама, у родитељској класи испред ознаке повратног типа метода наводи се нека од понуђених кључних речи.

Заокружити бројеве испред кључних речи које омогућавају редефинисање дефинисаног метода кроз ланац наслеђивања:

1. new
2. virtual
3. sealed
4. override
5. abstract
6. base
7. довољно је да буде public или protected

1,5

96. Дата је наредба кода у програмском језику C# која представља декларацију низа**.** Проценити које од доле наведених декларација су тачне.

Заокружити бројеве испред очекиваних одговора:

1. int niz = new int(30);
2. double[ ] niz = new double[30];
3. int[ ] niz = { 3, 4, 3, 2 };
4. char[ ] niz = new char[ ];
5. char[ ] niz = new char { 'a', 'b', 'c', 'd' };
6. char[ ] niz = new char[ ] { 'a', 'b' };

1,5

97. Дат је код на C#-у којим су креиране три класе у ланцу наслеђивања. Имајући у виду класификаторе приступа пољима класа, заокружити бројеве испред поља која ће бити видљива унутар класе Sin:

```
public class Deda {
   private double penzija;
   protected string adresa;
   public string ime;
}
public class Otac: Deda {
   private double plata;
   protected string struka;
}
public class Sin: Deda {
   public int razred;
}
```

1. penzija
2. adresa
3. ime
4. plata
5. struka
6. razred

1,5

98. Дат је код на C#-у којим су креиране три класе у ланцу наслеђивања. Унутар сваке класе декларисан је по један private, public и protected атрибут. У методи ***Main()*** класе ***Program*** креиран је објекат s класе ***Sin*** (`Sin s = new Sin();`) Заокружити бројеве испред поља која ће бити видљива у креираном објекту s класе Sin:

```
public class Deda {
   private double penzija;
   protected string adresa;
   public string ime;
}
public class Otac: Deda {
   private double plata;
    protected string  firma;
    public string struka;
}
public class Sin: Otac {
    private double prosek;
    protected int razred;
    public string skola;
}
```

1. penzija
2. adresa
3. ime
4. plata
5. struka
6. firma
7. prosek
8. razred
9. skola

1,5

99. Дата је дефиниција класе у програмском језику C# и састоји се од два конструктора, методе и поља x и y. У петом реду дефинисан је конструктор са параметрима који формира тачку са координатама x и y. Заокружити наредбе којима се може допунити дефиниција конструктора:

```
1. public class Point {
2.     private double x, y;
3.     public Point() { x = 0; y = 0; }
4.     public void set(double xx, double yy) { x = xx; y = yy; }
5.     public Point(double x, double y) { _____________; }
6. }
```

Заокружити бројеве испред тачних одговора:

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

2

100 Дати су делови кода у програмском језику C# који треба да рачунају збир елемената матрице **a,** декларисане на следећи начин: `int[,] a = new int[10, 10];`. Анализирати дате кодове и проценити који од предлога је тачан.

Заокружити бројеве испред очекиваних одговора:

1.
```
int sum = 0;
for (int i = 0; i < b.Length; i++)
                for (int j = 0; j < b[i].Length; j++)
                    sum3 += b[i][j];
```

2.
```
int sum = 0;
foreach (int x in a) sum1 += x;
```

3.
```
int sum = 0;
for (int i = 0; i < a.GetLength(0); i++)
                for(int j=0; j<a.GetLength(1); j++)
                    sum2 += a[i,j];
```

4.
```
int sum = 0;
foreach (int[] vrsta in b)
                foreach (int el in vrsta)
                    sum4 += el;
```

2

101 Дате су наредбе у програмском језику C# које дефинишу заглавље методе Print() са променљивим бројем параметара. Одредити који од понуђених одговора су исправни.

Заокружити бројеве испред очекиваних одговора:

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

2

102 Дата је дефиниција класе у програмском језику C# и састоји се од два конструктора, једне методе и поља x. У дефиницији се користи службена реч **this**. Анализирати дати код и проценити тачност следећих исказа. Заокружити бројеве испред тачних исказа:

```
class TestPrimer {
        public double x;
        public TestPrimer(double x){
            this.fun();
            this.x = x;
        }
        public TestPrimer(){
            Console.WriteLine("Podrazumevani konstruktor");
            this(23);
        }
        public void Fun(){
            Console.WriteLine("Poziv metoda fun()");
        }
    }
```

1. this.Fun() у конструктору TestPrimer(double x) може се поједноставити и заменити само са Fun().
2. this.x у конструктору TestPrimer(double x) може се поједноставити и заменити само са x.
3. позив конструктора this(23) унутар другог конструктора TestPrimer() је прво шта се извршава и мора се писати одмах после декларације `public TestPrimer():this(23)`
4. this(23) у конструктору Test() мора се заменити са прецизнијим изразом this(23.0).

2

103 Дати су искази који у програмском језику C# дефинишу конструктор.

Заокружити бројеве испред очекиваних одговора:

1. Подразумевани конструктор без параметара се увек аутоматски додаје класи.
2. Подразумевани конструктор без параметара се класи аутоматски додаје уколико у њој није експлицитно дефинисан ниједан конструктор.
3. У класи се мора експлицитно дефинисати бар један конструктор.
4. Конструктори немају тип резултата, чак ни **void.**

2

104 Заокружити бројеве испред наведених чланова класе који се ни под којим условима **НЕ** наслеђују са родитељске класе на изведену класу:

1. Readonly својства
2. Заштићени чланови класа
3. Својства (property)
4. Приватни чланови класа
5. Конструктор класе

2

105 Дат је код програма у програмском језику C#. Код садржи објекте две класе у којима је дефинисан метод ***ToString().*** Анализирати код датог програма и одредити који од датих исказа су тачни.

```
namespace TestPrimer {
    class Program {
        static void Main(string[] args) {
            Object a = new Klasa();
            Object obj = new Object();
            Console.WriteLine(a);
            Console.WriteLine(obj);
        }
    }
}
class Klasa{
   int x;
   public override string ToString() {return "x u A je " + x;}
}
```

2

Заокружити бројеве испред очекиваних одговора:

1. Програм има грешку, јер наредбу ***Console.WriteLine(a)*** треба заменити наредбом ***Console.WriteLine(a.ToString()).***
2. Приликом извршавања наредбе ***Console.WriteLine(a)***, програм позива се метод ToString() наслеђен из класе Object.
3. Приликом извршавања наредбе ***Console.WriteLine(a)***, програм позива метод ToString() из класе Klasa.
4. Приликом извршавања наредбе ***Console.WriteLine(obj)***, програм позива метод ToString() из класе Object.

106 У програмском језику C# дата је декларација две класе: ***KlasaA*** и ***KlasaB*** која наслеђује класу ***KlasaA***. Анализирати дате класе и проценити који од понуђених исказа су тачни.

```
namespace TestPrimer {
    class Program {
        static void Main(string[] args) {
            KlasaB b = new KlasaB();
            b.Print();
        }
    }
    class KlasaA {
        string s;
        public KlasaA(string s) { this.s = s; }
        public void Print() { Console.WriteLine(this.s); }
    }
    class KlasaB :KlasaA{ }
}
```

2

Заокружити бројеве испред очекиваних одговора:

1. Програм има грешку, јер ***KlasaB*** нема подразумевани конструктор ***KlasaB***().
2. Програм има грешку јер ***KlasaB*** има подразумевани конструктор, док родитељска ***KlasaA*** нема такав конструктор. Програм би радио без грешке уколико би се уклонио конструктор са параметрима из ***KlasaA***.
3. Програм има грешку која се може отклонити уколико би се у ***KlasaA*** експлицитно додао конструктор без параметара ***KlasaA***().
4. Програм нема грешку, извршава се, али се на конзоли ништа не исписује јер је поље ***s*** добило подразумевану вредност ***String.Empty***

107 У класи **Figura** дат је подразумевани (deafult) конструктор и конструктор са 4 параметра:

```
public Figura() {...}
public Figura(string ime, string boja, int pozX, int pozY) {...}
```

Заокружити бројеве испред исправно написаних наредби креирања објеката класе Figura:

1. `Figura f = Figura("lovac", "beli", 7, 3);`
2. `Figura f = new Figura("beli", "lovac", 7, 3);`
3. `Figura f = new Figura();`
4. `Figura f = new Figura("lovac", 3, 7, "beli");`
5. `Figura f = new Figura("lovac", "beli", 3, 7);`
6. `Figura f = new Figura("lovac", "beli", 3);`

3

Допуните следеће реченице и табеле

108. Дате су започете изјаве које се односе на делове кода за обраду изузетака. Довршити започете реченице:

Наредбе које се извршавају у случају настанка грешке стављају се унутар блока — catch

Наредбе које се извршавају и ако се деси и ако се не деси грешка стављају се унутар блока — finally

Наредбе које могу узазвати грешку стављају се унутар блока — try

1,5

109. На програмском језику C# дефинисане су две класе:

```
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(); }
}
```

Унутар функције Main, креирана су три објекта ових класа на следећи начин:

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

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

`r.Uvecaj();`	10
`rDin.Uvecaj();`	200
`rDev.Uvecaj();`	60

3

110. На програмском језику C# дефинисане су две класе:

```
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(); }
}
```

Унутар функције Main, креирана су три објекта ових класа на следећи начин:

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

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

`r.Uvecaj();`	10
`rDin.Uvecaj();`	200
`rDev.Uvecaj();`	250

3

111. На програмском језику C# дефинисане су две класе:

```
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; } }
```

Креирани су објекти ових класа и из њих позвана метода ***Metod().*** На предвиђене линије уписати шта метод ***Metod()*** враћа при позиву из наведених објеката:

`KlasaA a = new KlasaA(); a.Metod()` враћа вредност	10
`KlasaB b = new KlasaB(); b.Metod()` враћа вредност	20
`KlasaA bb = new KlasaB(); bb.Metod()` враћа вредност	20
`KlasaC c = new KlasaC(); c.Metod()` враћа вредност	30
`KlasaB cc = new KlasaC(); cc.Metod()` враћа вредност	20
`KlasaA ccc = new KlasaC(); ccc.Metod()` враћа вредност	20

3

112. На програмском језику C# дефинисане су две класе:

```
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"); }
}
```

Унутар функције Main креирана су два објекта ових класа на следећи начин:

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

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

`x.Poruka1();`	D1
`x.Poruka2();`	D2
`y.Poruka1();`	D1
`y.Poruka2();`	R2

4

У следећим задацима уредите и повежите појмове према захтеву

113. Са леве стране дате су кључне речи које одређују типове класа, а са десне су описи класа. На линију испред описа уписати редни број под којим је наведен одговарајући тип класе:

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

2

114. Са десне стране су наведене су области видљивости појединих елемената класе, а са леве стране класификатори приступа којима се врши контрола области видљивости. На линију испред описа области видљивости унети редни број под којим је наведен одговарајући класификатор приступа:

1.	private	3	видљив унутар класе у којој је дефинисан, као и унутар изведених класа
2.	public	1	видљив само унутар класе у којој је дефинисан
3.	protected	4	видљив унутар пројекта у коме је дефинисан
4.	internal	2	видљив и ван своје класе у којој је дефинисан

2

115. Са леве стране су наведени делови/елементи класе, а са десне стране улоге појединих класних елемената. На линију испред описа улоге унети редни број под којим је наведен одговарајући елеменат класе:

1.	поље (атрибут)	4	Опис функционалности објеката класе
2.	деструктор	5	Контрола приступа пољима класе
3.	конструктор	1	Опис особина објеката класе
4.	метод	3	Креирање објеката класе
5.	својство / property	2	Уништавање објеката класе

2,5

ВЕБ ДИЗАЈН

У следећим задацима заокружите број испред траженог одговора

161. Заокружити број испред исказа који тачно дефинише појам веб сајта:

1. скуп протокола за отпремање и преузимање података са интернета, као и протоколи за комуникацију на интернету
2. скуп веб-страница које могу да садрже текст, слике, видео снимке и други мултимедијални садржај састављен у једну целину
3. веб сајт чини интернет прегледач - програм који се користи за читање хипертекст докумената
4. интернет сервис (www) који омогућава корисницима размену докумената која се састоје од текста, слика и мултимедијалних компоненти

1

162. Заокружити број испред тачног одговора.

Основни протокол који користи веб сервер је:

1. SMTP
2. HTTP
3. HTML
4. WWW

1

163. Заокружити број испред тачног одговора. Која је основна улога CSS-a (Cascading Style Sheet):

1. Дефинише структуру и садржај странице
2. Дефинише конкретан изглед елемената странице
3. Дефинише којим језиком веб сервер извршава динамичку веб страницу
4. Дефинише протокол за приступ одређеној веб страници

1

164. Заокружити број испред тачног одговора.

Унутар HTML странице, CSS правила се креирају навођењем селектора елемента, својстава и вредности. Како се дефинишу **селектори** HTML елемента:

1. Селектори се дефинишу искључиво на основу имена HTML елемента
2. Селектори се дефинишу на основу имена елемента, назива класе или идентификатора елемента
3. Селектори се дефинишу на основу атрибута сваког HTML елемента
4. Селектори се дефинишу коришћењем кључне речи `selector` и атрибута

1

165. Заокружити број испред тачног одговора.

Дефинисан је стил елемента у оквиру кога се налази позадинска слика. Коју вредност својство `background-repeat` треба да има уколико слика не треба да се понавља у оквиру елемента:

1. `background-repeat: fixed;`
2. `background-repeat: null;`
3. `background-repeat: no-repeat;`
4. `background-repeat: repeat-x;`

1

166. Заокружити број испред тачног одговора.

Селектором ***a:visited*** омогућава се стилизовање:

1. активног линка
2. посећеног линка
3. линка када се преко њега пређе мишем
4. елемента са називом класе: a

1

167. Заокружити број испред тачног одговора.

Које од понуђених својстава омогућава постизање ефекта слојевитости елемената у оквиру HTML стране?

1. Својство `overlap`
2. Својство `layer`
3. Својство `z-index`
4. Не постоји решење за постизање слојевитости елемената HTML стране

1

168. Заокружити број испред тачног одговора.

Аналазирати следећи HTML код и одабрати која од понуђених опција описује резултат приказа HTML кода:

```
<b><i> Maturski ispit </i></b>
```

1. Текст ће бити само подебљан
2. Текст ће бити само искошен
3. Текст ће бити исписан подебљано и искошено
4. Неће се применити никакав ефекат на текст

1

169. Заокружити број испред линије кода којом се у HTML страницу укључује екстерна CSS датотека ***style.css*** са циљем дефинисања изгледа елемената дате странице:

1. <style type="text/css">....</style>
2. <link rel="stylesheet" type="text/css" href="style.css"/>
3. <body style="background-color:red;">...</body>
4. CSS датотека

2

170. Заокружити број испред тачног одговора. Који је од наведених начина за укључивање CSS стила најпогоднији за стилизовање, а посебно за касније одржавање и ажурирање HTML странице:

1. Увежена екстерна CSS датотека преко хипервезе у заглављу HTML веб странице - External Style Sheet
2. Дефинисан интерни CSS стил у оквиру head секције <style type="text/css">...</style> - Internal Style Sheet
3. Дефинисан стил у оквиру елемента HTML стране - Inline style
4. Не постоји решење за укључивање CSS стила у стилизовање HTML странице

2

171. Заокружити број испред тачног одговора.

Дефинисан је HTML елемент:

```
<div id="container"> Maturski ispit - EIT</div>
```

Које CSS правило дефинише стил за дати елемент:

1.
```
container {
        font-size: 1.5em;
         }
```
2.
```
#container {
        font-size: 1.5em;
          }
```
3.
```
.container {
        font-size: 1.5em;
          }
```
4.
```
selector container {
        font-size: 1.5em;
            }
```

2

172. Заокружити број испред тачног одговора.

Уколико је стил једног `DIV` елемента дефинисан са три класе `blue`, `black` и `top`, заокружити линију кода којом је елемент исправно дефинисан у оквиру HTML стране:

1. `<div id="blue black top">Maturski ispit -EIT</div>`
2. `<div id="blue" class="black" class="top">Maturski ispit -EIT</div>`
3. `<div class="blue" class="black" class="top">Maturski ispit - EIT</div>`
4. `<div class="blue black top">Maturski ispit -EIT</div>`

2

173. Заокружити број испред тачног одговора.

Дато је својство и додељена му је вредност:

background-attachment: fixed;

Закључити који од понуђених исказа описује дефинисани стил елемента са позадинском сликом:

2

1. Позадинска слика се помера са остатком садржаја на страни
2. Позадинска слика се не понавља у оквиру елемента
3. Позадинска слика је непомична (фиксирана) у односу на остатак садржаја
4. Позадинска слика се понавља унутар елемента

174. Заокружити број испред траженог одговора.

Дат је следећи код који стилизује три елемента странице различитим позадинским бојама:

```
<div style="background-color:red;
      width:300px;
      height:100px;
      position:relative;
      top:10px;
      left:80px;
      z-index:2">
</div>

<div style="background-color:yellow;
      width:300px;
      height:100px;
      position:relative;
      top:-60px;
      left:35px;
      z-index:1;">
</div>

<div style="background-color:green;
      width:300px;
      height:100px;
      position:relative;
      top:-220px;
      left:120px;
      z-index:3;">
</div>
```

2

Проценити шта од понуђеног описује ефекат извршења горњег кода:

1. Слој са зеленом позадином је на врху и преклапа остале слојеве
2. Слој са жутом позадином је на врху и преклапа остале слојеве
3. Слој са црвеном позадином је на врху и преклапа остале слојеве
4. Слојеви се утапају у једну боју – нијансу смеђе

У следећим задацима заокружите бројеве испред тражених одговора

175 Заокружите тачне исказе. Интернет је:

1. Интернет је светски систем умрежених рачунарских мрежа
2. Софтвер за преглед и приказ www страница се сматра Интернетом
3. Подаци који „путују“ светском мрежом и скуп корисника заједно чине Интернет мрежу
4. Интернет чини њена хардверска компонента као и систем софтверских слојева који контролишу различите аспекте њене комуникационе инфраструктуре

2

176 Заокружити бројеве испред тражених одговора.

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

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

2

177 Међу понуђеним исказима заокружити оне које се односе на динамичке веб странице:

1. Могу приступити базама података
2. Странице се састоје искључиво од HTML кода
3. Динамичке странице се пишу у CSS језику
4. Могу слати персонализован садржај појединачним корисницима
5. Странице се извршавају на веб серверу, а резултат овог извршавања представља HTML код
6. Ажурирање података на сајту је компликованије и спорије него код статичког сајта

3

178 Заокружити могуће начине укључивања CSS стила у оквиру HTML стране:

1. Увежена екстерна CSS датотека преко хипервезе у заглављу HTML веб странице - External Style Sheet
2. Дефинисан екстерни CSS стил приказивања у прегледачу HTML страница – CSS plugin
3. Дефинисан интерни CSS стил у оквиру head секције <style type=”text/css”>...</style> - Internal Style Sheet
4. Дефинисан стил у оквиру елемента HTML стране - Inline style
5. Дефинисан стил на крају HTML стране изван head и body секције – Outline style
6. Дефинисан CSS стил у оквиру секције body, унутар тага <css></css>

3

Допуните следеће реченице и табеле

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

<link rel=”stylesheet” type=”text/css” href=”style.css”/>

3

У следећим задацима уредите и повежите појмове према захтеву

180. Повезати појмове према захтеву.

Са леве стране су дати нумерисани HTML тагови, а са десне су наведени индикатори. На линију уписати број којим се HTML таг повезује са одговарајућим индикатором који ће се видети на HTML страници:

1.	<font color="#ff0003">	3	приказ наслова трећег нивоа
2.	<b>	4	приказ текста дате величине
3.	<h3>	1	приказ текста у боји
4.	<font size=„3">	2	приказ подебљаног текста

1,5

181. Повезати појмове према захтеву.

Са леве стране су дати нумерисани HTML тагови, а са десне су наведени индикатори. На линију уписати број којим се HTML таг повезује са одговарајућим индикатором који ће се видети на HTML страници:

1.		2	дефинисање елемента листе
2.		3	постављање позадинске боје
3.	<body bgcolor="#ffff00">	4	постављање позадинске слике
4.	<body background="0001.jpg">	1	дефинисање нумерисане листе

1,5

182. Повезати појмове према захтеву.

Са леве стране су дати нумерисани HTML тагови, а са десне су наведени индикатори. На линију уписати број којим се HTML таг повезује са одговарајућим индикатором који ће се видети на HTML страници:

1.	<tr>	4	Дефинисање хиперлинка
2.	<td>	1	Нови ред у ћелији
3.	<img src="slika.jpg" >	2	Нова ћелија у табели
4.	<a href="index.html" >	3	Уметање слике

183. Са леве стране су дати интернет сервиси, а са десне стране су дефинисане операције које се извршавају помоћу тих сервиса.

На линију испред дефинисане операције написати број њему одговарајућег сервиса.

1. E-mail	4	успостављање везе са удаљеним рачунаром и рад на њему
2. FTP	3	приказ HTML веб страница
3. WWW	1	слање електронске поште
4. Telnet	2	пренос датотека са удаљеног сервера

2

184. Повезати појмове према захтеву.

HTML документ може да прими податке од корисника помоћу форми (формулара). Повежите дате тагове и атрибуте са њиховим дефиницијама:

1. FORM	3	Дефинише где проследити податке са форме после предаје (submit) форме
2. INPUT	4	Одређује начин на који се подаци са форме шаљу на дефинисано одредиште (може бити „post” или „get“)
3. ACTION	1	Основни таг формулара са којим се креира формулар за унос података од стране корисника
4. METHOD	2	Дефинише поље за унос податка унутар HTML форме.

2

185. Направите редослед потребних корака за приказ статичке веб странице уносом редних бројева на линије испред описа корака, почев од броја 1 до броја 5.

4	Веб сервер шаље пронађену страну клијенту - веб претраживачу.
2	Корисник захтева да види веб страну (кликом на линк, укуцавањем адресе у адресну линију веб претраживача и слично)
1	Аутор је креирао страну која се састоји од HTML кода и ставио је на веб сервер.
5	Веб претраживач обрађује добијени HTML код и приказује кориснику уредно форматирану страницу са свим елементима (сликама, линковима, табелама, …)
3	Веб сервер проналази HTML страну коју је корисник захтевао

3

186. Одредите редослед потребних корака за приказ динамичке веб странице уносом редних бројева (почев од 1) на линије испред описа корака.

6	Веб претраживач обрађује добијени HTML код и приказује кориснику страницу са свим елементима.
4	Веб сервер извршава програмски код који је саставни део стране и креира HTML код.
2	Корисник креира захтев за преглед динамичке веб стране. Захтев се прослеђује од клијента (веб претраживача) до веб сервера на коме се налази захтевана страна.
1	Аутор је креирао страну која се састоји од серверских контрола и инструкција у неком програмском језику и ставио је на веб сервер.
3	Веб сервер обрађује захтев корисника и проналази динамичку страну коју је корисник захтевао
5	Веб сервер шаље преко Интернета генерисани HTML код веб претраживачу.

3

187. Уредите појмове према захтеву.

Дат је низ HTML наредби. Уписивањем редног броја на предвиђену линију (почев од броја 1 до броја 6) поређајте тачним редоследом HTLM наредбе у складу са основном структуром HTML странице:

5	</body>
2	<head>
6	</html>
1	<html>
3	</head>
4	<body>

ВЕБ ПРОГРАМИРАЊЕ

У следећим задацима заокружите број испред траженог одговора

188.	Заокружити исказ који описује улогу Domain Name Server-а: 1. превођење имена домена у IP адресу 2. хостовање веб сајта 3. вршење функције главног чвора у локалној рачунарској мрежи 4. приказ динамичких веб страница	**1**
189.	Једну од платформи за развој веб апликација развио је и Microsoft. Заокружити назив Microsoft-ове платформе за развој веб апликација: 1. HTML 2. JSP 3. PHP 4. ASP.NET	**1**
190.	ASP.NET MVC 3.0 долази са новом техником за дефинисање погледа (View Engine). Заокружити назив ове технике: 1. ASP.NET View Engine 2. Salome 3. Razor 4. Default	**1**
191.	Зокружити од понуђених опција који сумбол се користи за коментаре у ASP.NET MVC Razor синтакси: 1. // 2. /* … */ 3. <!--...--> 4. @* … *@	**1**
192.	SOAP протокол (Simple Object Access protocol) који се користи за размену података између рачунара коришћењем веб сервиса, у основи користи један скрипт језик. Заокруживањем редног броја, обележити о ком језику се ради. 1. HTML 2. CSS 3. JavaScript 4. XML	**1**

193.	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	**2**
194.	Заокружити исказ који допуњује опис понашања променљивих током извршења програма написаног у језику JavaScript. *Током извршавања апликације написане у JavaScript-у*: 1. Није могуће мењати типове променљивих у току извршавања апликације 2. Типови променљивих се могу мењати током извршавања програма 3. Типови променљивих се обавезно мењају током извршавања апликације у одговарајући веб тип променљиве 4. JavaScript не подржава типове промељивих	**2**
195.	Дата је ASP.NET MVC апликација у којој је креирана нова мастер страница (master layout page) која се зове *_Layout.WindowsPhone.cshtml*. Ако желимо да укључимо нову мастер страницу на новом погледу (View) који сегмент кода треба да искористимо. Заокружити тачан одговор: 1. @Html.ActionLink("_Layout.WindowsPhone.cshtml"); 2. Layout = "~/views/Shared/_layout.WindowsPhone.cshtml"; 3. Layout = "Layout.WindowsPhone.cshtml"; 4. @Html.Partial("_Layout.WindowsPhone.cshtml");	**2**
196.	Заокружити један од понуђених типова сервиса који може бити хостован у конзолној или десктоп апликацији: 1. ASMX 2. RESTful 3. WCF 4. XML	**2**

У следећим задацима заокружите бројеве испред тражених одговора

197.	Заокружите број испред назива скрипт језика за израду динамичких веб страница: 1. PHP 2. jQuery 3. ASP.NET 4. JSP 5. VBScript 6. CSS 7. HTML	**1,5**

198. У клијентском скрипт језику Java Script постоји неколико могућности конверзије стринга у број. Заокружити функције које омогућавају те конверзије: 1. Није могуће мењати типове променљивих у току извршавања апликације 2. Функција EVAL - процењује стринг и ако је могуће претвара га у број 3. Функција parseINT – конвертује стринг у целобројни број, ако је могуће 4. Функција parseFLOAT – конвертује стринг у реалан број, ако је могуће 5. Функција parseDOUBLE – конвертује стринг у реалан број, ако је могуће 6. Функција tryParseINT – конвертује стринг у целобројни број, ако је могуће	**1,5**
199. Обележити шта од наведеног је дефинисано WSDL језиком (Web Services Description Language) (заокружити више понуђених одговора): 1. Комуникациони интерфејс за веб сервис 2. Начин импелементације метода веб сервиса 3. Списак метода веб сервиса 4. Комуникациони протокол за веб сервис	**2**
200. Microsoft .NET Framework садржи базне класе које пружају широк спектар могућности. Заокружити елементе који су укључени у .NET Framework: 1. класе корисничког интерфејса 2. класе за приступ подацима и базама 3. класе корисника 4. веб сервер и примере базе података 5. класе за манипулацију XML докумената 6. скрипт језик који се извршава на клијент страни	**3**
201. Повратне вредности акције MVC контролера (controller action method) могу бити различитих типова. Заокружити бројеве испред могућих повратних вредности. 1. ViewResult 2. MVCResult 3. ModelResult 4. JsonResult 5. RedirectResult 6. ASPResult	**3**
202. XML Schema (XSD) опсује структуру XML документа Обележити шта од наведеног дефинише XML Schema (заокружити више понуђених одговора): 1. Типове података XML елемената и атрибута 2. Вредности XML елемената и атрибута 3. XML елементе који представљају child (деца) елементе 4. Уређење child елемената 5. Начин сортирања атрибута у оквиру елемената 6. Међусобни однос корених (root) елемената документа	**3**

Допуните следеће реченице и табеле

203. Дата је MVC стандардна рута (default route)

```
http://localhost/Customer/Details/5
```

која има 3 сегмента. Препознати на основу дате руте вредности ових сегмената и допунити реченицу:

Име контролера (Controller Name) је: _Customer_, назив методе (Action Method Name) је:

Details а ID параметра методе је дат са: _5_.

1,5

У следећим задацима уредите и повежите појмове према захтеву

204. У JavaScript језику свака веб страница има објекте наведене у левој колони. На десној страни дати су описи наведених објеката. Повезати објекат са одговарајућим описом уносом редног броја којим је објекат нумерисан на означену линију:

1. window	3	садржи URL адресе које су претходно биле посећене, као и URL адресе које су посећене након посете тренутној страници
2. location	4	представља тренутно учитани документ и служи као приступна тачка садржају странице
3. history	2	садржи информације о URL адреси документа који је тренутно приказан у посматраном прозору
4. document	1	објекат највишег нивоа који садржи све друге објекте и представља прозор претраживача

2

205. У JavaScript језику многи објекти имају уграђене функције (методе) које симулирају догађаје настале услед акција корисника. Уносом редног броја методе (лева колона) на предвиђену линију, повежите метод и одговарајућу акцију корисника.

1. focus	4	Изађе из фокуса елемента форме
2. submit	3	Учита страницу у прегледач
3. load	1	Уђе у фокус неког елемента форме
4. blur	2	Изврши слање форме

2

206. Веб обрасци могу да садрже различите типове компоненти. Редни број под којим је наведена категорија компоненти уписати на линију испред одговарајуће групе компоненти веб образаца:

1. HTML контроле	3	TextBox, Label, Button, ListBox, DropDownList, DataGrid
2. Контроле за податке	4	FileSystemWatcher, EventLog, MessageQueue
3. Серверске контроле	2	SqlConnection, SqlCommand, OleDbConnection
4. Системске компоненте	1	Text Area, Table, Image, Submit Button, Reset Button

4

207. Садашње верзије IIS сервера изграђене су на модуларној архитектури. Повежите називе модула са функцијама које обављају:

1. HTTP модули	3	Модули за праћење и извештавање о догађајима насталим током процесирања захтева...
2. Безбедностни модули	4	Модули за обраду захтева за статичке фајлове, враћање подразумевне странице када клијент не наведе ресурс у захтеву, излиставање садржаја директоријума...
3. Модули за евиденцију и дијагностику	1	Модули за одговорарање на информације, враћање HTTP грешака, преусмеравање захтева...
4. Модули садржаја	2	Модули за ауторизацију УРЛ адреса, филтрирање захтева...

4

У следећим задацима заокружите број испред траженог одговора

208. Међу понуђеним ентитетима, одредити ентитет са атрибутима који **НИСУ** одговарајући. Заокружити број испред траженог одговора:

1. Ентитет: СТУДЕНТ – Атрибути: име, презиме, смер, број бодова, просек
2. Ентитет: ДРЖАВА – Атрибути: назив, број становника, површина
3. Ентитет: КЊИГА – Атрибути: наслов, аутор, година издања, издавач, адреса издавача, телефон издавача
4. Ентитет: АВИОН – Атрибути: произвођач, марка, година производње, број седишта

1

209. Заокружите тачан исказ:

1. Примарни кључ је атрибут који мора бити целобројног типа.
2. Примарни кључ је атрибут који указује на зависност од неке друге табеле.
3. Примарни кључ је атрибут који јединствено идетификује врсте у табели.
4. Ако табела садржи вишевредносни атрибут, њему се додељује функција примарног кључа.

1

210. Заокружити број испред траженог одговора.

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. LIKE
3. BETWEEN
4. IS LIKE
5. BEGINS WITH

1

211. Заокружити број испред траженог одговора.

Табела UCENICI поред осталих података, садржи и вредност стипендије. Одредити оператор који треба употребити у WHERE клаузули SELECT наредбе да би били приказани сви ученици код којих није позната и није унета вредност у колону ***stipendija***:

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

1

212. Дат је упит:

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

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

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

1

213. 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. HAVING клаузулу
2. GROUP BY клаузулу
3. JOIN клаузулу
4. ORDER BY клаузулу

1

214. Заокружити тачан исказ:

1. Кардиналност неке везе представља однос броја објеката који се повезују.
2. Кардиналност неке везе представља апстракцију у којој се скуп сличних типова објеката представља општим генеричким типом (надтипом).
3. Кардиналност неке везе одређује опционалност учешћа у вези.
4. Кардиналност показује колико кандидата за примарни кључ има неки тип ентитета.

2

215. Дата је табела *PROJEKAT* над којом се извршава упит:

```
ALTER TABLE PROJEKAT ADD RokKraj date
```

Одредити шта ће се десити након извршења упита. Заокружити број испред траженог одговора.

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

2

216. Дата је табела ***RADNIK*** и упит:

IDBR	IME	PREZIME	PLATA	PREMIJA	DATZAP
6234	Marko	Pavlović	1300	3000	1990-12-17
6789	Janko	Nikolić	3900	10	1999-12-23

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

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

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 краја века
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 тренутног датума

2

217. Дата је табела ***RADNIK***, табела ***ODELJENJE*** и упит:

IDBR	IME	PREZIME	PLATA	BROD
5900	Slobodan	Golubović	900	10
5932	Mitar	Gavrilović	600	10
5953	Persida	Kosanović	1100	20
6234	Marko	Pavlović	1300	30
6789	Janko	Nikolić	800	10

BROD	IMEOD	MESTO
50	Skladišta	Zemun
30	Marketing	Vračar
10	Plasman	Surčin
20	Direkcija	Grocka
40	Nabavka	Barajevo

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

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

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

2

218. Дата је табела ***RADNIK***, табела ***ODELJENJE*** и упит:

IDBR	IME	PREZIME	PLATA	BROD
5900	Slobodan	Golubović	900	10
5932	Mitar	Gavrilović	600	
6234	Marko	Pavlović	1300	30
6789	Janko	Nikolić	800	10

BROD	IMEOD	MESTO
50	Skladišta	Zemun
30	Marketing	Vračar
10	Plasman	Surčin
20	Direkcija	Grocka

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

Одредити шта се види као резултат датог упита. Заокружити број испред траженог одговора:

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

2

219. Дата је табела ***RADNIK***, табела ***ODELJENJE*** и упит:

IDBR	IME	PREZIME	PLATA	BROD
5900	Slobodan	Golubović	900	10
5932	Mitar	Gavrilović	600	10
5953	Persida	Kosanović	1100	20
6234	Marko	Pavlović	1300	30
6789	Janko	Nikolić	800	10

BROD	IMEOD	MESTO
50	Skladišta	Zemun
30	Marketing	Vračar
10	Plasman	Surčin
20	Direkcija	Grocka
40	Nabavka	Barajevo

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

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

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

2

220. Извршава се следећа SELECT наредба:

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

Заокруживањем броја испред одговарајућег исказа, одредити које ће вредности бити приказане:

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.

2

221. 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 > (SELECT AVG(cena) FROM proizvod)
```

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

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

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

2

222. Извршава се упит:

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

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

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

2

223. Табела ***ARTIKLI*** садржи следеће колоне: *artikl_id, naziv, kategorija, cena, kolicina*.

Потребно је да се прикаже категорија и минимална цена артикла у свакој категорији. При томе се тражи приказ само оних категорија где је најмања цена производа већа од задате граничне вредности која се преноси упиту кроз параметар *@granica*

Изабрати упит који даје тражени извештај:

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
   GROUP BY kategorija
   ```

2

224. Табела ***RADIONICA*** садржи следеће колоне: ***radionica_id, naziv, zanat, lokacija_id***.

Потребно је да се прикаже колико на свакој локацији има различитих заната. Заокружити број испред упита који даје тражени извештај:

1. ```
   SELECT DISTINCT location_id, COUNT(zanat)
   FROM radionica
   GROUP BY lokacija_id
   ```
2. ```
   SELECT location_id, COUNT(zanat)
   FROM radionica
   GROUP BY lokacija_id
   ```
3. ```
   SELECT location_id, COUNT(DISTINCT zanat)
   FROM radionica
   GROUP BY lokacija_id
   ```
4. ```
   SELECT location_id, COUNT(DISTINCT zanat)
   FROM radionica
   GROUP BY zanat
   ```

2

225. Заокружити број испред одговора који представља наставак датог исказа:

Уколико поглед (view) треба користити за измену података у табели, поглед **НЕ СМЕ** садржати...

1. WHERE клаузулу
2. Спој више табела
3. Алијас колоне
4. GROUP BY клаузулу

2

226. Дат је упит за креирање погледа и наведени искази који се односе на дати упит. 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 се могу само у додавати у табелу ***Testovi***, али не и мењати
3. Овако дат упит изазива грешку при извршењу
4. Коришћењем датог погледа, подаци из табеле ***Testovi*** се могу само прегледавати, али не и додавати или мењати

2

У следећим задацима заокружите бројеве испред тражених одговора

227 Заокружити бројеве испред тражених одговора. Међу понуђеним алатима, обележити ***Case*** алате:

1. Rational Rose
2. Oracle Designer
3. .NET
4. Microsoft Visio
5. Java
6. SQL Express

1,5

228 Обележити команде које се сматрају командама ажурирања података у бази података. Заокружити бројеве испред тражених одговора:

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

1,5

229 Заокружити бројеве испред тражених одговора.

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 ASC за растући редослед
2. и службена реч DESC за опадајући редослед
3. и службена реч ASC за опадајући редослед
4. и службена реч DESC 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

1,5

230 Заокружити бројеве испред наредби које служе за креирање, брисање и измену структуре релационе базе и објеката који чине релациону базу: 1. ALTER TABLE 2. INSERT 3. CREATE TABLE 4. DROP TABLE 5. UPDATE 6. DELETE TABLE 7. ADD COLUMN 8. ADD CONSTRAINT	**1,5**
231 Заокружити бројеве испред тражених одговора. Одредити ентитете који садрже одговарајуће атрибуте: 1. Ентитет: КЊИГА – Атрибути: наслов, аутор, издавач, година издања 2. Ентитет: АУТОМОБИЛ – Атрибути: марка, година производње, боја, власник, година рођења власника, регистарски број 3. Ентитет: УЧЕНИК – Атрибути: име, презиме, разред, одељење, број оправданих, број неоправданих, просек 4. Ентитет: ДРЖАВА – Атрибути: назив, број становника, површина, главни град, број становника главног града, име градоначелника главног града	**2**
232 Дати су искази који се односе на спољашњи кључ табеле (*foreign key constraint*). Заокружити бројеве испред **<u>тачних</u>** исказа: 1. Вредност у пољу спољашњег кључа не сме бити NULL 2. Вредност спољашњег кључа мора бити јединствена (unique) у колони над којом је постављено ограничење спољашњег кључа 3. Вредност у пољу спољашњег кључа мора бити или NULL или једнака некој од вредности из колоне на коју спољашњи кључ референцира 4. Више редова у табели може садржати исту вредност у пољу спољашњег кључа и тиме показивати на исти ред у референцираној табели 5. Колона спољашњег кључа не мора садржати исти тип података као колона на коју спољашњи кључ референцира	**2**
233 Заокружити бројеве испред тражених одговора. Означити операторе који се **<u>НЕ МОГУ</u>** користити за поређење са подупитом који враћа више вредности: 1. ALL 2. ANY 3. BETWEEN 4. SOME 5. LIKE 6. IN	**2**

234 Дата је табела ***Kupci*** са структуром:

```
( Id int primary key, Prezime varchar(50), Adresa varchar(50), Mesto
varchar(20), Telefon varchar(5), Status varchar(8) )
```

И табела ***NoviKupci*** са структуром:

```
( Id int primary key, Prezime varchar(50), Telefon varchar(20),  Status
varchar(8) )
```

Извршава се упит:

```
INSERT INTO NoviKupci
SELECT * FROM Kupci WHERE Status <> 'Aktivan'
```

2

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

1. Како табела ***NoviKupci*** има све колоне које постоје и у табели ***Kupci,*** упит се извршава без грешке и у табелу ***NoviKupci*** се уписују записи из табеле **Kupci** са статусом који није **Aktivan**
2. Упит се не извршава, пријављује грешку јер се број колона у табели ***Kupci*** разликује од броја колона у табели ***NoviKupci***
3. Упит би се извршио без грешке да су у SELECT клаузули подупита, уместо * наведене све колоне табеле ***Kupci*** које имају своју одговарајућу колону у табели ***NoviKupci***
4. Упит јавља грешку због покушаја уписа вредности у поље примарног кључа који је аутоматски, тј креира га сама база

235 Дата је табела ***RADNIK***, табела ***ODELJENJE*** и упит:

IDBR	IME	PREZIME	PLATA	SIFRAOD
5900	Slobodan	Golubović	900	10
5932	Mitar	Gavrilović	600	
5953	Persida	Kosanović	1100	20

SIFRAOD	IMEOD	MESTO
10	Marketing	Vračar
20	Direkcija	Grocka
30	Nabavka	Barajevo

Креирана је ускладиштена процедура са параметром `@br int = NULL`

Позивом процедуре извршава се упит:

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

2

Одредити који од понуђених исказа су тачни. Заокружити бројеве испред тражених одговора:

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 прераспоређени у одељење чији је brod 30

236 Табела **Zaposleni** садржи поља: Zaposleni_Id, Ime, Prezime, Plata, Odsek_Id.

Дат је упит:

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

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

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

2

237 Заокружити бројеве под којима су наведене клаузуле SQL наредбе у којима се могу користити аритметичке операције:

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

2

238 Извршава се следећи упит:

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

Одредити које две вредности може вратити ова наредба. Заокружити бројеве испред тражених вредности:

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

2

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

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

2

240 Креиране су табеле ***SKOLA*** и ***OSNOVNASKOLA***, а затим су у табелу ***SKOLA*** уписани подаци извршавањем следећих наредби:

```
create table Skola(
      skolaID int primary key, Naziv nvarchar(50),  gradID int, tip
nvarchar(50) )
create table OsnovnaSkola(
      gimID int primary key, Naziv nvarchar(50), gradID int )

insert into Skola values (101,'Nikola Tesla',20, 'srednja strucna')
insert into Skola values (102,'Dusko Radovic', 20,'osnovna')
insert into Skola values (103,'Sveti Sava', 30,'osnovna')
insert into Skola values (104,'Bora Stankovic', 20,'gimnazija')
```

У табелу ***OSNOVNASKOLA*** треба уписати податке о основним школама преписивањем потребних вредности из табеле ***SKOLA.*** Заокружити бројеве испред упита који ће јавити грешку при извршењу: **2**

1.
```
select * into OsnovneSkole
from Skola where tip='osnovna'
```
2.
```
insert into OsnovneSkole
select * from Skola where tip='osnovna'
```
3.
```
insert into OsnovneSkole(skolaID, Naziv)
select s.skolaID, s.Naziv from Skola as s where tip='osnovna'
```
4.
```
insert into OsnovneSkole
select s.skolaID, s.Naziv, s.gradID from Skola as s where
tip='osnovna'
```

241 Креиране су и попуњене подацима табеле ***Korisnik*** и ***Prijatelji.*** Њихова структура и садржај приказани су на слици:

Korisnik

ID	IME	POL
1	Ana	NULL
2	Steva	m
3	Marta	z
4	Petra	z

Prijatelji

Korisnik1	Korisnik2
1	2
1	3
2	3

Извршавањем упита добија се табела са подацима.

```
select k.ime, COUNT(*) as [broj prijatelja]
from Korisnik as k
left join Prijatelji as p on p.korisnik1=k.id or p.korisnik2=k.id
where k.pol='z'
group by k.id, k.ime
```

2

Заокруживањем бројева испред понуђених одговора, обележити који од наведених података ће бити приказани у појединим редовима резултујуће табеле:

1. Ana, 1
2. Ana, 2
3. Steva, 1
4. Steva, 2
5. Marta, 1
6. Marta, 2
7. Petra, 0
8. Petra, 1

242	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	**2**
243	Одредити ентитете који садрже одговарајуће атрибуте. Заокружити бројеве испред тражених одговора: 1. Ентитет: СТУДЕНТ – Атрибути: име, презиме, смер, број бодова, просек 2. Ентитет: КЊИГА – Атрибути: наслов, аутор, година издања, издавач, адреса издавача, телефон издавача 3. Ентитет: АВИОН – Атрибути: произвођач, марка, година производње, број седишта 4. Ентитет: ДРЖАВА – Атрибути: назив, број становника, површина 5. Ентитет: САЈАМ – Атрибути: назив, број излагача, покровитељ, адреса покровитеља, контак особа покровитеља 6. Ентитет: ТУРИСТИЧКА АГЕНЦИЈА – Атрибути: назив, адреса, година оснивања, власник, стручна квалификација власника, запослени, стручна квалификација запослених	**3**
244	Обележити ентитете код којих је извршен адекватан избор јединственог идентификатора. Заокружи бројеве испред тражених одговора: 1. јединствени матични број грађанина (ЈМБГ) за ентитет ОСОБА 2. датум рођења за ентитет ОСОБА 3. ИСБН број за ентитет КЊИГА 4. регистарска ознака за АУТОМОБИЛ 5. дестинација за ентитет АРАНЖМАН 6. режисер за ентитет ФИЛМ	**3**
245	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 све вредности подупита задовољавају услов 3. Оператор ANY врши поређење са свим вредностима које враћа подупит и враћа TRUE ако било која од вредности подупита задовољава услов 4. Оператору ANY може да претходи оператор LIKE или оператор IN. 5. Оператору ANY мора да претходи оператор поређења (=, <>, >, >=, <, <=) 6. Услов =ANY(*скуп вредности*) је еквивалентан услову IN (*скуп вредности*)	**3**

246 Креирана је табела ***SKOLA***, а затим су у њу уписани подаци извршавањем следећих наредби:

```
create table Skola( skolaID int primary key, Naziv varchar(50) )

insert into Skola values (101,'Nikola Tesla')
insert into Skola values (102,'Mihajlo Pupin')
insert into Skola values (103,'ETS Zemun')
```

За дати упит, треба проценити сценарио који ће се десити и заокруживањем редних бројева испред исказа, означити могуће исходе:

```
select * into StrucneSkole from Skola
```

1. Креира се копија табеле Skola - нова табела под именом StrucneSkole исте структуре као и табела Skola и у њу се преписују сви подаци из табеле Skola
2. Уколико табела са именом StrucneSkole постоји у бази, креира се нова са именом StrucneSkole(1) и у њу се преписују сви редови из табеле Skola
3. Уколико табела са именом StrucneSkole постоји у бази, не креира се нова, само се у постојећу преписују редови из табеле Skola
4. Уколико табела са именом StrucneSkole постоји у бази, упит јавља грешку
5. Уколико се дода услов `where 1=2` упит се извршава, креира се нова табела исте структуре као и табела Skola, али се у њу не уписује ни један ред
6. Уколико се дода услов `where 1=2` упит јавља грешку јер је 1=2 увек нетачно тј. ***False***
7. Упит јавља грешку јер се кључна реч `into` користи искључиво у комбинацији са `insert`

3

Допуните следеће реченице и табеле

247. Допунити реченицу наводећи назив нормалне форме:

Уколико ни један атрибут релације није вишевредносни, нити композитни, тј. не може се раставити, кажемо да је релација у _____ првој _____ нормалној форми.

1

248. Допунити реченицу наводећи назив нормалне форме:

Уколико сви атрибути релације који нису део кључа зависе од сваког атрибута који је део кључа кажемо да је релација у ___ другој _____ нормалној форми.

1

249. Допунити реченицу наводећи назив нормалне форме:

Уколико сви некључни (споредни) атрибути релације не зависе од неког другог некључног атрибута, тј. ако не постоји транзитивна зависност било ког споредног атрибута од било ког кључа те релације, кажемо да је релација у ____ трећој _____ нормалној форми.

1

У следећим задацима уредите и повежите појмове према захтеву

250. 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

1,5

251. Дата је табела ***GEOGRAFIJA*** која поред осталих података садржи називе градова и држава (*Naziv nvarchar(50)*). У зависности од услова у ***WHERE*** клаузули, ***SELECT*** упитом се приказују географски појмови из табеле. Са леве стране су дати услови нумерисани бројевима од 1 до 5, а са десне групе градова.

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%'` — 5 — LA VALETA, LA KORUNJA
3. `where Naziv like '% L%'` — 2 — EL RENO, LA KORUNJA
4. `where Naziv like '_L%'` — 4 — EL SALVADOR, EL RENO
5. `where Naziv like '__ %A'` — 1 — LAS VEGAS, LOS ANGELES

2,5

252. Исписати на цртама испред релација редни број под којим је наведена одговарајућа кардиналност везе:

1. 1 : 1
2. 1 : M
3. M : M

2 — ВЛАСНИК – БРОЈ ТЕЛЕФОНА
3 — НАСТАВНИК – ПРЕДМЕТ
1 — ОСОБА – ПАСОШ
3 — КУПАЦ – МОДЕЛ АУТОМОБИЛА
2 — УТАКМИЦА – ГРАД ДОМАЋИН

2,5

253. Уписати редни број почев од 1 на линију испред резервисане речи тако да одговара редоследу навођења при формирању упита.

За формирање упита за издвајање дела података из табеле која се налази у оквиру базе података користе се клаузуле у следећем редоследу:

4 — GROUP BY
3 — WHERE
1 — SELECT
5 — ORDER BY
2 — FROM

3

254. Дата је табела ***RADNIK***, табела ***ODELJENJE*** и упит:

IDBR	IME	PREZIME	PLATA	BROD
5900	Slobodan	Golubović	900	10
5932	Mitar	Gavrilović	600	
5953	Persida	Kosanović	1100	20
6234	Marko	Pavlović	1300	30
6789	Janko	Nikolić	800	10

BROD	IMEOD	MESTO
50	Skladišta	Zemun
30	Marketing	Vračar
10	Plasman	Surčin
20	Direkcija	Grocka
40	Nabavka	Barajevo

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
```
3

2.
```
SELECT odeljenje.imeod,
radnik.prezime
FROM odeljenje LEFT JOIN radnik
ON radnik.brod = odeljenje.brod
```
1

3.
```
SELECT odeljenje.imeod,
radnik.prezime
FROM odeljenje RIGHT JOIN
radnik
ON radnik.brod = odeljenje.brod
```
2

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

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

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

3

255. Дата је табела ***RADNIK***, табела ***ODELJENJE*** и упит:

IDBR	IME	PREZIME	PLATA	BROD
5900	Slobodan	Golubović	900	10
5932	Mitar	Gavrilović	600	
5953	Persida	Kosanović	1100	20
6234	Marko	Pavlović	1300	30
6789	Janko	Nikolić	800	10

BROD	IMEOD	MESTO
50	Skladišta	Zemun
30	Marketing	Vračar
10	Plasman	Surčin
20	Direkcija	Grocka
40	Nabavka	Barajevo

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
```

3.
```
SELECT odeljenje.imeod,
radnik.prezime
FROM odeljenje FULL JOIN radnik
ON radnik.brod = odeljenje.brod
```

1.
```
SELECT odeljenje.imeod,
radnik.prezime
FROM odeljenje RIGHT JOIN
radnik
ON radnik.brod = odeljenje.brod
WHERE odeljenje.brod IS NULL
```

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

3

256. На левој страни су наведене категорије SQL команди, а са десне су набројане команде. 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

3	GRANT
2	UPDATE
4	COMMIT
1	DROP
2	DELETE
1	ALTER

3

257. Табела ***ZAPOSLENI*** је креирана и попуњена извршавањем следећих наредби:

```
create TABLE zaposleni(
  id INTEGER NOT NULL PRIMARY KEY, rukovodilacId INTEGER, ime VARCHAR(30)
NOT NULL,
  FOREIGN KEY (rukovodilacId) REFERENCES zaposleni(id))

INSERT INTO zaposleni VALUES(1, NULL, 'Petar');
INSERT INTO zaposleni VALUES(2, 1, 'Mihajlo');
INSERT INTO zaposleni VALUES(3, 2, 'Milica');
INSERT INTO zaposleni VALUES(4, 3, 'Lazar');
INSERT INTO zaposleni VALUES(5, NULL, 'Sofija');
```

Очекивани ефекти извршења упита нумерисани су бројевима од 1 до 6. Уносом редног броја одговарајућег описа на предвиђену линију испред упита, повезати упит и опис резултата његовог извршења:

1. Сви радници који су руководиоци неком другом раднику
2. Сви радници који нису руководиоци ником
3. Сви радници који немају надређене руководиоце
4. Сви радници који имају надређеног руководиоца
5. Празна табела

5	`select * from zaposleni where id not in (select distinct rukovodilacId from zaposleni)`
2	`select * from zaposleni where id not in (select rukovodilacId from zaposleni where rukovodilacId is not null)`
3	`select * from zaposleni where rukovodilacId is null`

3