# Computer Science Graduation Exam 2018
# Practice Questions - Discrete Structures and Algorithms

Note:

*The written test of the graduation exam (July or September 2018) will consist of 60 questions which will be similar (in structure and difficulty level) with those included in this booklet. For each of the three categories (Discrete Structures and Algorithms, Programming Languages and Software Engineering, Computing Systems and Databases) there will be 20 questions.*

*If there are unclear aspects concerning the questions and/or answers please contact the teacher(s) who proposed the questions for each section:*

**Algorithms:**

- Gabriel Istrate (gabriel.istrate@e-uvt.ro)

**Logic for Computer Science:**

- Adrian Crăciun (adrian.craciun@e-uvt.ro)

**Data Structures:**

- Cosmin Bonchiş (cosmin.bonchis@e-uvt.ro)

**Graph Theory and Combinatorics:**

- Isabela Drămnesc (isabela.dramnesc@e-uvt.ro)

**Formal Languages and Automata Theory:**

- Mircea Drăgan (mircea.dragan@e-uvt.ro)

## 1 Algorithms

1. Let us consider the algorithm:

   ```
   a:=1
   WHILE a<>n DO
     a=2*a
   ENDWHILE
   ```

   For which values of n the loop is NOT infinite?

   (a) For any natural nonzero value
   (b) For any natural nonzero value which is even
   (c) For any natural power of 2
   (d) For any even power of 2
   (e) For any odd power of 2

2. Let us consider the algorithm:

   ```
   a:=0
   FOR i:=1,n DO
     i:=i-h
     a:=a+1
   ENDFOR
   ```

   For which values of h the loop is not finite?

   (a) for h=0
   (b) for any h which is smaller than 0
   (c) for none value of h
   (d) for any h larger or equal to 1

3. Let us consider the algorithm:

   ```
   c:=a MOD q
   a:=a DIV q
   WHILE a>0 DO
      IF c<a MOD q THEN c:=a MOD q ENDIF
      a:=a DIV q
   ENDWHILE
   ```

   Knowing that a has a natural value, q has a natural value between 2 and 10, the value of c after the execution of the above algorithm will be:

(a) The least significant digit from the representation of `a` in base `q`

(b) The most significant digit from the representation of `a` in base `q`

(c) The smallest digit from the representation of `a` in base `q`

(d) The largest digit from the representation of `a` in base `q`

(e) The largest multiple of `q` which divides `a`

4. Let us consider an array `b[0..n]` containing the binary digits of a natural value (`b[n]` corresponds to the most significant digit and `b[0]` corresponds to the least significant digit). Let us suppose that exists at least one index `i` for which `b[i]=0`.

```
alg(b[0..n])
  i:=0
  WHILE b[i]=1 DO
    b[i]:=0
    i:=i+1
  ENDWHILE
  b[i]:=1
  RETURN b[0..n]
```

Which of the following statements is(are) true for algorithm `alg`:

(a) `alg` transforms in 0 all elements of `b[0..n]` which are equal to 1

(b) the complexity order of `alg` is $\Theta(n)$

(c) the complexity order of `alg` is $\mathcal{O}(n)$

(d) `alg` increments the value having `b[0..n]` as representation in base 2

5. Let us consider an array `b[0..n]` containing the binary digits of a natural value (`b[n]` corresponds to the most significant digit and `b[0]` corresponds to the least significant digit). Let us suppose that exists at least one index `i` for which `b[i]=1`.

```
alg(b[0..n])
  i:=0
  WHILE b[i]=0 DO
    b[i]:=1
    i:=i+1
  ENDWHILE
  b[i]:=0
  RETURN b[0..n]
```

Which of the following statements is(are) true for the algorithm `alg`:

(a) the complexity order of `alg` is $\mathcal{O}(n)$

(b) the complexity order of `alg` is $\Theta(n)$;

(c) `alg` decrements the value having `b[0..n]` as its base 2 representation

(d) `alg` transforms in 1 all elements of `b[0..n]` which are equal to 0

6. Let us consider the algorithm:

```
d:=m
i:=n
r:=d MOD i
WHILE r<>0 DO
  d:=i
  i:=r
  r:=d MOD i
ENDWHILE
```

For which values of `m` and `n`, the variable `i` will contain the value of the greatest common divisor of `m` and `n`?

(a) for m and n natural values which satisfy `m<n`

(b) for m and n nonzero natural values which satisfy `m>n`

(c) for any nonzero natural values `m` and `n`

(d) for none value

7. Let us consider the algorithm:

```
FOR d:=2,n DO
  IF n MOD d=0 THEN
    WRITE d
    WHILE (n MOD d=0) DO
      n:=n DIV d
    ENDWHILE
  ENDIF
ENDFOR
```

For a natural number $n \geq 2$, the execution of the algorithm will print:

(a) All positive divisors of `n`

(b) All divisors of `n`

(c) All natural divisors of `n` which are prime numbers

(d) All natural divisors of `n` which are odd numbers

(e) None value

8. Let `x[1..n]` be an array with integer values and the algorithm:

```
FOR i:=1,m DO
  a:=x[i]
  x[i]:=x[n-i+1]
  x[n-i+1]:=a
ENDFOR
```

   What value should have `m` such that by executing this algorithm the order of elements in `x` is reversed and the number of swaps is as small as possible?

   (a) n
   (b) n DIV 2
   (c) n DIV 2 +1
   (d) 3n DIV 2
   (e) None value of `m` can ensure that the order of elements of `x` is reversed

9. Let us consider an array `x[1..n]` and the algorithm:

```
k1:=1
k2:=1
FOR i:=2,n DO
    IF x[k1]>x[i] THEN k1:=i
                  ELSE IF x[k2]<=x[i] THEN k2:=i ENDIF
    ENDIF
ENDFOR
```

   Which of the following statements is(are) true after the execution of the algorithm?

   (a) `x[k1]`$\leq$`x[k2]`
   (b) `x[k1]`$\geq$`x[k2]`
   (c) `k1` is the last position which contains the minimum of `x` and `k2` is the first position containing the maximum of `x`
   (d) `k1` is the first position which contains the minimum of `x` and `k2` is the last position containing the maximum of `x`
   (e) `k1` is the first position which contains the maximum of `x` and `k2` is the last position containing the minimum of `x`
   (f) `k1` is the last position which contains the maximum of `x` and `k2` is the first position containing the minimum of `x`

10. Let `x[1..n]` be an array with real values and the following algorithm:

```
i:=1
j:=n
WHILE i<j DO
  WHILE (x[i]<0) AND (i<n) DO i:=i+1 ENDWHILE
  WHILE (x[j]>0) AND (j>1) DO j:=j-1 ENDWHILE
  IF i<j THEN
    aux:=x[i]
    x[i]:=x[j]
    x[j]:=aux
  ENDIF
ENDWHILE
```

What is the effect of the algorithm on `x`?

(a) It sorts `x` in an increasing order

(b) It removes all positive elements from `x`

(c) It removes all negative elements from `x`

(d) It moves all negative elements before the positive ones (the index of any negative element will be smaller than the index of any positive element)

(e) It moves all positive elements before the negative ones (the index of any positive element will be smaller than the index of any negative element)

11. Let us consider a squared matrix, `a[1..n,1..n]`, with elements from $\{0, 1\}$ and the algorithm:

```
alg (a[1..n,1..n])
FOR r=1,n-1 DO
  s1:=0; s2:=0;
  FOR i:= r+1,n DO
    s1:=s1+a[i,i-r]
    s2:=s2+a[i-r,i]
  ENDFOR
  IF NOT(s1=1) OR NOT(s2=1) THEN RETURN False ENDIF
ENDFOR
RETURN True
```

Which of the following statement(s) is(are) true ?

(a) The algorithm returns `True` in all cases when each row and each column of the matrix contains exactly one value equal to 1

(b) The algorithm returns `True` only when the main diagonal and all the diagonals parallel to it contain exactly one value equal to 1

(c) The body of the `FOR` loop on `i` is executed for $n(n + 1)/2$ times

(d) The algorithm returns `False` only if all diagonals parallel to the main diagonal contain only values equal to 0

(e) The body of the `FOR` loop on `i` is executed for $n(n-1)/2$ times

12. Let us consider the algorithm:

```
FOR k:=0,n-1 DO
  FOR j:=1,n-k DO
     WRITE a[j+k,j]
  ENDFOR
ENDFOR
```

The number of printed values is:

(a) $n^2/2$

(b) $n + n^2/2$

(c) $n(n-k)$

(d) $n(n+1)/2$

13. Let us consider the algorithm:

```
FOR i:=1,n-1 DO
   IF x[i]>x[i+1] THEN
     aux:=x[i]
     x[i]:=x[i+1]
     x[i+1]:=aux
   ENDIF
ENDFOR
```

The complexity order of the algorithm (with respect to the number of swaps) is:

(a) $\mathcal{O}(\lg n)$

(b) $\mathcal{O}(n)$

(c) $\mathcal{O}(n \lg n)$

(d) $\mathcal{O}(n^2)$

(e) $\Theta(n)$

14. Let `x[1..n]` be an increasingly sorted array ($n > 1$), `v` a value and the algorithm:

```
s:=1
d:=n
r:=0
```

```
WHILE (s<=d) AND (r=0) DO
    m:=(s+d) DIV 2
    IF v=x[m] THEN r:=1
              ELSE IF v<x[m] THEN d:=m-1
                             ELSE s:=m+1
                   ENDIF
    ENDIF
  ENDWHILE
```

Which of the following statement(s) is(are) true?

(a) The minimal number of comparisons involving `v` is 2

(b) The complexity order with respect to the number of comparisons is $\mathcal{O}(n)$

(c) The complexity order with respect to the number of comparisons is $\mathcal{O}(\lg n)$

(d) The minimal number of comparisons involving `v` is 1

(e) The variable `r` contains the value 1, if `v` is in the array and 0 otherwise

(f) The variable `r` contains the value 1, if `v` is in the array and 0 otherwise

15. Let `x[1..n]` be an array. Which is the complexity order of the following algorithm ?

```
nr:=0
i:=1
WHILE i<=n DO
  k:=0
  WHILE (x[i+k]>0) AND ((i+k)<n) DO k:=k+1 ENDWHILE
  IF nr<k THEN nr:=k ENDIF
  i:=i+k+1
ENDWHILE
```

(a) $\mathcal{O}(n^2)$

(b) $\mathcal{O}(n)$

(c) $\mathcal{O}(\log n)$

(d) $\mathcal{O}(n \cdot k)$

(e) $\mathcal{O}(n \cdot \log k)$

16. Let `x[1..n]` be an unsorted array. We are looking for a pair, `(imax, jmax)`, of indices from $\{1, 2, \ldots, n\}$ which has the property that the absolute value `|x[imax]-x[jmax]|` is maximal. Which of the following algorithms finds a correct pair by using $\mathcal{O}(n)$ operations?

(a)
```
imax:=1; jmax:=2
FOR i:=1,n-1 DO
FOR j:=i+1,n DO
IF |x[i]-x[j]|>|x[imax]-x[jmax]|
THEN imax:=i; jmax:=j
ENDIF
ENDFOR
ENDFOR
```

(b)
```
imax:=1; jmax:=1
FOR i:=2,n DO
   IF x[imax]>x[i] THEN imax:=i ENDIF
   IF x[jmax]<x[i] THEN jmax:=i ENDIF
ENDFOR
```

(c)
```
imax:=1; jmax:=1
FOR i:=2,n DO
   IF x[imax]<x[i] THEN imax:=i ENDIF
   IF x[jmax]>x[i] THEN jmax:=i ENDIF
ENDFOR
```

17. Let us consider an increasingly sorted array, `a[1..n]`, a value `x` and the problem of checking if there exists at least one pair `(a[i],a[j])` (with distinct indices `i` and `j`) satisfying `a[i]+a[j]=x`.

```
alg1 (a[1..n],x)
  i:=1; j:=n
  WHILE i<j DO
    IF a[i]=x-a[j] THEN RETURN True
       ELSE IF a[i]<x-a[j] THEN i:=i+1
              ELSE j:=j-1
           ENDIF
    ENDIF
  ENDWHILE
  RETURN False

alg2(a[1..n],x)
  FOR i:=1,n DO
    FOR j:=1,n DO
      IF a[i]+a[j]=x THEN RETURN True
    ENDFOR
  ENDFOR
  RETURN False

alg3(a[1..n],x)
  FOR i:=1,n-1 DO
    FOR j:=i+1,n DO
```

```
        IF a[i]+a[j]=x THEN RETURN True
        ELSE RETURN FALSE
      ENDFOR
    ENDFOR
```

Which of the following statements is (are) true?

(a) The algorithm `alg1` solves correctly the problem and its complexity order is $\mathcal{O}(n)$

(b) The algorithm `alg3` has a complexity order smaller than `alg2`

(c) The algorithm `alg2` solves correctly the problem and its complexity order is $\mathcal{O}(n^2)$

(d) The algorithm `alg3` solves correctly the problem and its complexity order is $\mathcal{O}(n^2)$

18. Let us consider the algorithm:

```
k:=0
i:=1
WHILE i<=n DO
   k:=k+1
   i:=2*i
ENDWHILE
```

The number of multiplications executed by this algorithm is:

(a) $\mathcal{O}(n)$

(b) $\mathcal{O}(n^2)$

(c) $\mathcal{O}(\lg n)$

(d) $\mathcal{O}(2n)$

(e) $\mathcal{O}(n/2)$

19. Let us consider the algorithm:

```
s:=0
m:=1
FOR i:=1,n DO
   m:=2*m
   FOR j:=1,m DO s:=s+1 ENDFOR
ENDFOR
```

The number of incrementations of variable `s` in the above algorithm is:

(a) $\mathcal{O}(n)$

(b) $\mathcal{O}(n^2)$

(c) $\mathcal{O}(n \lg n)$

(d) $\mathcal{O}(2^n)$

(e) $\mathcal{O}(n^4)$

20. Let us consider the following algorithms to compute the factorial of a natural number `n`:

```
fact1(n)
  f:=1
  FOR i:=2,n DO
    f:=f*i
  ENDFOR
  RETURN f
```

```
fact2(n)
IF n<=1 THEN RETURN 1
        ELSE RETURN n*fact2(n-1)
ENDIF
```

Which of the following statements is(are) true? (the multiplication is considered as dominant operation)

(a) The complexity order of `fact1` is higher than the complexity order of `fact2`

(b) The complexity order of `fact2` is higher than the complexity order of `fact1`

(c) The algorithms `fact1` and `fact2` have the same complexity order

21. Let us consider the algorithm:

```
alg(n)
  IF n>0 THEN
    alg(n DIV 2)
    WRITE n MOD 2
  ENDIF
```

If `alg` is called for a nonzero input parameter, `n`, it will print:

(a) The binary digits of `n` (starting with the least significant digit)

(b) The binary digits of `n` (starting with the most significant digit)

(c) The remainder of the division of `n` by 2

(d) The quotient of the division of `n` by 2

22. Let us consider the algorithm:

```
alg(n)
  IF n=0 THEN RETURN 0
         ELSE RETURN n MOD 2+alg(n DIV 2)
  ENDIF
```

Supposing that the algorithm is called for a nonzero natural number `n`, which of the following statements is(are) true?

(a) The algorithm returns the number of binary digits of `n`

(b) The algorithm returns the number of digits equal to 1 in the binary representation of `n`

(c) The algorithm returns the sum of the digits from the binary representation of `n`

(d) The algorithm returns the number of digits equal to 0 in the binary representation of `n`

23. Let us consider the following algorithm (which has access to `x[1..n]`):

```
alg(i, j)
  IF i<j THEN
    aux:=x[i]
    x[i]:=x[j]
    x[j]:=aux
    alg(i+1,j-1)
  ENDIF
```

The effect of `alg(1,n)` is:

(a) It interchanges the first element of the `x` with the last one

(b) It reverses the order of elements in `x[1..n]`

(c) It does not change `x`

(d) It interchanges the elements of `x` which are neigbours, i.e. `x[i]` is interchanged with `x[i+1]` for each `i` in $\{1, 2, \dots, n-1\}$

24. Let us consider an array `x[1..n]` and the algorithm:

```
sort(x[1..n])
  FOR i:=2,n DO
    aux:=x[i]
    j:=i-1
    WHILE (j>=1) AND <condition> DO
       x[j+1]:=x[j]
       j:=j-1
    ENDWHILE
    x[j+1]:=aux
  ENDFOR
```

Which of the following expressions should be placed instead of ¡condition¿ such that the algorithm sorts `x` increasingly?

(a) `x[j]<=aux`

(b) `x[j]>=aux`

(c) `x[j]<aux`

(d) `x[j]>aux`

25. Let us consider the increasingly sorted array, `x[1..n]`, a value `v` and the algorithm:

```
  alg(x[1..n],v)
    g:=0
    i:=1
    WHILE (g=0) AND (i<=n) DO
      IF v<=x[i] THEN g:=1
                 ELSE i:=i+1
ENDIF
    ENDWHILE
    RETURN i
```

Which of the following statements is(are) true?

(a) The algorithm has a linear complexity

(b) At the end of the `WHILE` loop, `g` has value 0 if and only if `v>x[n]`

(c) The algorithm returns the number of elements of `x` which are larger than `v`

(d) The algorithm returns the position where `v` can be inserted such that `x` remains increasingly sorted

(e) The algorithm always returns `(n+1)`

26. Let us consider an array `x[1..n]` and the algorithm:

```
Alg(x[1..n])
  FOR i:=n,2,-1 DO
    FOR j:=1,i-1  DO
       IF x[j]<x[j+1] THEN
         aux:=x[j]
         x[j]:=x[j+1]
         x[j+1]:=aux
       ENDIF
     ENDFOR
   ENDFOR
```

Which of the following statements is(are) true?

(a) The algorithm sorts `x[1..n]` increasingly

(b) The algorithm sorts `x[1..n]` decreasingly

(c) The algorithm does not sort the algorithm (neither increasingly nor decreasingly);

(d) The algorithm has a linear complexity disregarding the initial order of elements in `x` (i.e. the complexity order of the algorithm is $\Theta(n)$);

(e) The algorithm has a quadratic complexity disregarding the initial order of elements in `x` (i.e. the complexity order of the algorithm is $\Theta(n^2)$)

27. Let us consider a real value `x`, a nonzero natural value `n` and the algorithm:

```
alg(x,n)
  IF n=1 THEN RETURN x
         ELSE
           p:=alg(x,n DIV 2)
           IF n MOD 2=0 THEN RETURN p*p
                        ELSE RETURN p*p*x
   ENDIF
  ENDIF
```

Which of the following answers is true?

(a) The algorithm returns $x^2$ if `n` is even and $x^3$ if `n` is odd

(b) It is possible that the sequence of recursive calls does never end

(c) The algorithm returns $x^n$ if n is even and $x^{n+1}$ if `n` has an odd value

(d) The algorithm returns $x^n$ for any nonzero natural value of `n`

(e) The algorithm has linear complexity ($\mathcal{O}(n)$)

(f) The algorithm has logarithmic complexity ($\mathcal{O}(\log n)$).

28. Let us consider two natural values `a` and `b` and the algorithm:

```
alg(a,b)
  IF a=0 OR b=0 THEN RETURN 0
                ELSE
                  IF a MOD 2=0 THEN RETURN (alg(a DIV 2,2*b))
                               ELSE RETURN (alg(a DIV 2,2*b)+b)
                  ENDIF
  ENDIF
```

Which of the following statements is(are) true?

(a) It is possible that the sequence of recursive calls does never end

(b) The algorithm returns always 0

(c) The algorithm returns the product `a*b` if `a` is even and `a*b+b` if `a` is odd

(d) The algorithm returns the product `a*b` for any natural value `a`

29. Let us consider the algorithm `alg` called for a natural value `n` and let us denote by $T(n)$ the number of executed additions.

```
alg(n)
  IF n<=9 THEN RETURN n
          ELSE RETURN alg(n DIV 10)+n MOD 10
  ENDIF
```

Which of the following statements is(are) true?

(a) The algorithm returns the number of nonzero digits of `n`

(b) The algorithm returns the sum of digits of `n`

(c) $T(n) = 1$ if $n \leq 9$ and $T(n) = T(nDIV10) + nMOD10$ if $n > 9$

(d) $T(n) = 0$ if $n \leq 9$ and $T(n) = T([n/10]) + 1$ if $n > 9$

(e) $T(n)$ belongs $\mathcal{O}(n)$

(f) $T(n)$ belongs lui $\mathcal{O}(\log n)$

30. Let us suppose that the execution time of a recursive algorithm, used to solve a problem of size $n$, satisfies the following recurrence relation:

$T(n) = 1$ if $n = 1$, and $T(n) = nT(n-1) + 2$ if $n > 1$

Which is the complexity order of $T(n)$?

(a) $\mathcal{O}(2^n)$

(b) $\mathcal{O}(n!)$

(c) $\mathcal{O}(n^2)$

(d) $\mathcal{O}(2n)$

## 2 Logic for Computer Science

1. Consider the predicate logic language that contains the following symbols:

   - variables, indicated with lower case letters
   - function symbols $\mathcal{F}$: $+$ binary infix, $-$ unary prefix, $*$ binary infix.
   - predicate symbols $\mathcal{P}$: $=$,$<$, $\leq$ all binary, infix.
   - constant symbols $\mathcal{C}$: $0, 1$.

   Which of the following are terms over this language?

   (a) $(0 * x) - 1$,

   (b) $1 + (z * x) < 0$,

   (c) $x + ((-1) * 0)$,

   (d) $0 * (y + 1)$.

2. Consider the predicate logic language that contains the following symbols:

   - variables, indicated with lower case letters
   - function symbols $\mathcal{F}$: $+$ binary infix, $-$ unary prefix, $*$ binary infix.
   - predicate symbols $\mathcal{P}$: $=$,$<$, $\leq$ all binary, infix.
   - constant symbols $\mathcal{C}$: $0, 1$.

   Which of the following are formulae over this language?

   (a) $\forall x \forall y((x \leq 1) \Rightarrow (1 = 0))$,

   (b) $(x = y) \wedge (1 + 0)$,

   (c) $(x < (1 + 0)) \vee \forall x \exists y(x = (-y))$,

   (d) $\forall x((x - 1) = y) \wedge (x < 0)$.

3. Which of the following are inductive domains:

   (a) natural numbers,

   (b) integer numbers,

   (c) propositional logic formulae,

   (d) lists of integers.

4. Using the definition of well formed propositional formulae (wffs), decide which of the following are propositional formulae:

   (a) $(((P \to Q) \vee S) \leftrightarrow T)$,

   (b) $(P \to (Q \wedge (S \to T)))$,

   (c) $(\neg(B(\neg Q)) \wedge R)$.

5. For the following propositional formulae, and for the truth valuation $\{P, \neg Q\}$:
   (a) $((P \to Q) \wedge ((\neg Q) \wedge P))$ evaluates to $\mathbb{T}$,
   (b) $((P \to Q) \to (Q \to P)$ evaluates to $\mathbb{T}$,
   (c) $((\neg(P \vee Q)) \wedge (\neg Q))$ evaluates to $\mathbb{F}$.

6. Which of the following statements are true:
   (a) if a propositional formula is valid then it is satisfiable,
   (b) if a propositional formula is not valid then it is satisfiable,
   (c) if a propositional formula is not valid then its negation is satisfiable,
   (d) if a propositional formula is not valid, then its negation is valid.

7. What is the relation between propositions

$$(F \wedge G) \to H$$

   and

$$F \to (G \to H).$$

   (a) they are logically equivalent,
   (b) the first one is a logical consequence of the second one,
   (c) the second one is a logical consequence of the first one,
   (d) they are not related in any of the ways above.

8. The formula:

$$P \leftrightarrow Q$$

   is

   (a) logically equivalent to the conjunction of,
   (b) a logical consequence of,
   (c) logically equivalent to the disjunction of

   the formulae:

$$\begin{aligned} &Q \to R, \\ &R \to (P \wedge Q), \\ &P \to (Q \vee R). \end{aligned}$$

9. Which of the following formulae are in Disjunctive Normal Form?
   (a) $P$,
   (b) $\neg P \vee Q$,

(c) $P \wedge \neg Q \wedge S$,

(d) $(P \wedge \neg Q \wedge S) \vee \neg S$.

10. Which of the following are Disjunctive Normal Forms of the formula

$$(P \to Q)?$$

(a) $\top$,

(b) $\bot$,

(c) $\neg P \vee Q$,

(d) $(\neg P \wedge \neg Q) \vee (\neg P \wedge Q) \vee (P \wedge Q)$.

11. What is a resolvent of clauses $\{P, \neg Q, R\}$ and $\{\neg P, Q, S\}$?

(a) $\emptyset$,

(b) $\{P, \neg P, R, S\}$,

(c) $\{R, S\}$.

12. To establish whether a formula $G$ is a logical consequence of formulae $F_1, \ldots, F_n$, which of the following methods can be applied:

(a) check that $(F_1 \wedge \ldots \wedge F_n) \to G$ is unsatisfiable,

(b) check that $\neg F_1 \vee \ldots \vee \neg F_n \vee G$ is unsatisfiable,

(c) check that $\neg F_1 \vee \ldots \vee \neg F_n \vee G$ is valid,

(d) check that $F_1 \wedge \ldots \wedge F_n \wedge \neg G$ is unsatisfiable.

13. There is a formula which is logically equivalent to

$$\left(\begin{array}{c} ((P_1 \to (P_2 \vee P_3) \wedge (\neg P_1 \to (P_3 \vee P_4)))) \\ \wedge \\ ((P_3 \to (\neg P_6)) \wedge (\neg P_3 \to (P_4 \to P_1))) \\ \wedge \\ (\neg(P_2 \wedge P_5)) \wedge (P_2 \to P_5) \end{array}\right) \to \neg(P_3 \to P_6).$$

which contains only propositional connectives taken from:

(a) $\{\neg, \vee\}$,

(b) $\{\vee, \wedge\}$,

(c) $\{|\}$,

(d) $\{\bot, \to\}$.

where $|$ is the NAND connective (i.e. $P|Q = \neg(P \wedge Q)$).

14. Which of the following methods in propositional logic is a reasoning method (i.e. satisfies the properties of reasoning):
    (a) truth tables,
    (b) resolution,
    (c) the Davis Putnam method.

15. Consider the ternary boolean function which implements parity: it returns $\mathbb{T}$ if the number of $\mathbb{T}$ inputs is odd, $\mathbb{F}$ otherwise. Which of the following propositional formulae corresponds to (describes) this function?
    (a) $(P \wedge Q \wedge R) \rightarrow (P \vee Q \vee R)$,
    (b) $(\neg P \wedge \neg Q \wedge R) \vee (\neg P \wedge Q \wedge \neg R) \vee (P \wedge \neg Q \wedge \neg R) \vee (P \wedge Q \wedge R)$,
    (c) $(\neg P \rightarrow (Q \wedge R)) \vee (\neg R \rightarrow (P \wedge Q)) \vee (\neg Q \rightarrow (P \wedge R))$.

16. The clause set corresponding to the formula

$$(\neg P \rightarrow (Q \wedge R)) \rightarrow (P \rightarrow \neg Q)$$

    is:
    (a) $\{\{\neg P, \neg Q\}\}$,
    (b) $\{\{P, \neg Q\}, \{P, R\}, \{\neg Q, R\}\}$,
    (c) $\{\{P, \neg Q, \neg R\}, \{P, Q, R\}, \{\neg P, \neg Q, R\}\}$.

17. Consider the clause set containing the following clauses:

$$\begin{array}{ll} (1) & \{P, Q, \neg R\}, \\ (2) & \{\neg P, R\}, \\ (3) & \{P, \neg Q, S\}, \\ (4) & \{\neg P, \neg Q, \neg R\}, \\ (5) & \{P, \neg S\}. \end{array}$$

    The formula corresponding to this clause set is:
    (a) valid,
    (b) satisfiable,
    (c) unsatisfiable.

18. The Davis-Putnam method returns the answer satisfiable:
    (a) when the empty clause is generated,
    (b) when the empty clause set is generated,
    (c) when no new clauses can be generated, and the empty clause is not in the clause set.

19. Consider the clause set

$$\{\{F, \neg G, H\}, \{\neg F, \neg G\}, \{G, H\}, \{\neg F, H\}, \{F, G, \neg H\}\}.$$

    Which of the following rules of the Davis-Putnam-Logemann-Loveland method can be applied to it?

    (a) the pure literal rule,
    (b) the 1 literal rule,
    (c) the splitting rule.

20. To prove a goal $G$ when a disjunction $A \vee B$ is known:

    (a) assume $A$ and prove $G$ then assume $B$ and prove $G$,
    (b) assume $A$ and prove $G$,
    (c) assume $\neg A$ and prove $B$ and $G$.

## 3 Data Structures

1. From a physical point of view, an elementary data is defined as being:
   (a) a triplet of the form: (identifier, attributes, values)
   (b) a memory zone of a certain length stored at a certain absolute address in which there are memorized in time and in a shape specific to the given value
   (c) symbol through which a certain information is pointed during the running time

2. The components of a data structure:
   (a) must all be of the same type
   (b) can be themselves data structures
   (c) can be all of the same type
   (d) cannot be user defined types
   (e) can have one of the standard types of the used programming language

3. Select which of the following operations are considered basic operations on lists in general:
   (a) sorting increasingly/decreasingly a list's elements by the value of certain fields
   (b) summing up of a list's elements
   (c) accessing of the i'th element in a list
   (d) removal of a node in the list
   (e) copying of a list

4. Consider the following C code:

```
#include <stdio.h>
#include <conio.h>
#define lg_max 20
typedef int node;
typedef struct
  {
    node element[lg_max];
    int last;
  }list;

list l;
int k; /* k is an index in the list */

void initializare(void)
```

```
    {
     l.last=-1;
   } /* initialization */

  void insp(void)
   {
    int i;
    if(l.last>=lg_max-1) printf("Oversize\n");
    else
     {
      for (i=l.last;i>=k;i--)
        l.element[i+1]=l.element[i];
      printf("Input the value of a new element(integer):");
      scanf("%d",&l.element[k]);
      printf("\n");
      l.last++;
     }
   } /* insp */
```

The function insp() is useful when we want to:

(a) insert an element at the end of the list

(b) insert a node in the list

(c) delete the node from position k from the list

(d) find in the list a node of key ”k”

5. Consider the following C code:

```
#include <stdio.h>
#include <conio.h>
#define lg_max 20
typedef int node;
typedef struct
  {
    node elements[lg_max];
    int last;
  }list;

list l;

int i; /* i is an index in the list */
```

```
void initialization(void)
   {
     l.last=-1;
   } /* initialization */
```

What do you understand by deleting the node on position i, different from the last?

(a) the element on position i will take the value 0

(b) the last field is decremented, then all fields after position i are moved towards the end of the list

(c) all elements after position i are moved one step closer towards the beginning of the list, then the value of the last field is decremented

(d) all elements before the i position are moved towards the beginning of the list, then the value of the last field is decremented

6. A simply linked linear list is:

   (a) a linear list whose ordering relation is based on the existence of a pointer to the next element

   (b) a data structure implemented with the array type

   (c) an explicit primitive structure

7. Giving the C code:

```
  struct node
    {
       int key;
       char info[10];
       struct node *next;
    };
  typedef struct node Tnode;
  typedef Tnode * ref;
  ref p=NULL,q;
      /* p is a pointer to the first element of the list   */
```

And the function:

```
void func(void)
     {
        q=(ref)malloc(sizeof(Tnod e));
        printf("Input the key=  ");scanf("%d", &q->key);
        printf("Input the info=  ");
        fflush(stdin);scanf("%s", q->info);
        q->urm=p;
        p=q;
     }
```

We can say that the function will:

(a) create a simple linked list assigned by p with insertion in the front of the list;

(b) create the first node of the list;

(c) create a list with elements in reverse order of the given key;

(d) insert a new node in front of the list assigned by p;

(e) create a list that inserts in the back of the list.

8. Giving the C code:

```
 struct node
    {
       int key;
       char info[10];
       struct nod *next;
    };
  typedef struct node Tnode;
  typedef Tnode * ref;
  ref p; /* p - pointer to the first element of the list  */
  int k;
```

And the function:

```
void search(void)
  {
     int b = 0;
     ref q=p;
     while ((b==0) && ( q != NULL))
       if (q->key ==k)
          b=1;
       else q=q->next;
 }
```

which searches for a node of given key k into a simple linked list.The variable b must be introduced in this function for:

(a) dereferencing of a node of address NULL;

(b) avoid dereferencing a node of address NULL;

(c) to cover the list easier;

(d) to establish if the node exists or not in the list.

9. Having p a pointer to the first node of a simple linked list of integer numbers with the structure:

```
struct node
    {
   int info;
   struct node *next;
    };
  typedef struct node Tnode;
  typedef Tnode *ref;
  ref p,r;
```

Which will be the index of the r pointer after the execution of the following program:

```
int i=1;
r=p;
while(i<=5)
  {r=r->next;
   i++;
  }
```

(a) 3;

(b) 4;

(c) 5;

(d) 6;

(e) the sequence is wrong

10. Let p be a pointer to the first node of a simple linked list of integer numbers with the structure:

```
struct node
  {
     int info;
     struct node *next;
  };
typedef struct node Tnode;
typedef Tnode *ref;
ref p,r;
```

and the list contains in order the following numbers: 1, -2, 6,-3, -6. Specify what will display the next sequence:

```
  int i=0;
  r=p;
  while(r)
    {
     if(r->info==6)i++;
     r=r->next;
    }
  printf("%d",i)
```

(a) 0;
(b) 1;
(c) 2;
(d) 3;
(e) 6.

11. Let r and q two pointers which addresses two nodes of a simple linear linked list of integer numbers with the structure:

```
struct node
  {
    int info;
    struct node *next;
  };
typedef struct node Tnode;
typedef Tnode *ref;
ref r,q;
```

Which of the following sequences of instructions makes that the node pointed by q to be the successor of the node pointed by r, if the list has at least two nodes and r doesn't point to the last node of the list?

(a) r=q;

(b) q=r;

(c) q-¿next=r;

(d) r-¿next=q;

(e) r-¿next=q-¿next.

12. Let p be a pointer to the first node of simple linear linked list, of integer numbers, with the structure:

```
struct node
  {
    int info;
    struct node *next;
  };
typedef struct node Tnode;
typedef Tnode *ref;
ref p;
```

Which of the following instructions display the number contained in the third node of the list?

(a) `printf("%d",p->next->next->info);`

(b) `printf("%d",p->next->info);`

(c) `printf("%d",p->next->info->info);`

(d) `printf("%d",p->next->next->next);`

(e) `printf("%d",p->next->next->next->info);`

13. Let p and q be two pointers which point to the first elements of two simple linear linked list of integer numbers with the structure:

```
struct node
  {
    int info;
    struct node *next;
  };
typedef struct node Tnode;
typedef Tnode *ref;
ref p,q;
```

What will the following function do?

```
void test(ref p,ref q)
  {
     int sw,aux;
     ref r,u;
     r=p;
     while(r)
       {
          sw=0;
          aux=r->info;
          u=q;
          while(u)
             {
                if(aux==u->info)
                   sw=1;
                u=u->next;
             }
           if(sw)
             printf("%d",aux);
           r=r->next;
       }
  }
```

(a) displays the elements of the first list that are not found in the second one;

(b) displays the elements from the second list that are not found in the first one;

(c) displays the common elements from the two lists;

(d) displays only the distinct elements from the lists;

(e) none of the above.

14. Let consider the descriptions in C language:

```
struct node
  {
    int info;
    struct node *next;
  };
typedef struct node Tnode;
typedef Tnode *ref;
ref p,r;
/* p points to the first node of the list,
   r points to the node that will  be erased*/
```

Which of the following sequences describe(s) the deletion of the first node of a simple linked linear list, when this is the first and the last node of the list?

(a)
```
ref s;
  s=p;
  while (s->next!=r)
         s=s->next;
  s->next=0;
  free(r);
```

(b)
```
ref s;
  s=r->next;
  *r=*s;
  free(s);
```

(c)
```
p=NULL;
  free(r);
```

15. Let the descriptions in C language:

```
struct node
  {
    int info;
    struct node *next;
  };
typedef struct node Tnode;
typedef Tnode *ref;
ref p,r;
/* p points to the first node of the list,
   r points to the node that will  be erased*/
```

Which of the following sequences describe the deletion of the last node of a simple linked linear list, when this is not the first node of the list?

(a)
```
s=p;
  while (s->next!=r)
         s=s->next;
  s->next=0;
  free(r);
```

(b)
```
s=r->next;
  *r=*s;
  free(s);
```

(c)
```
p=NULL;
 free(r);
```

16. Let p be a pointer to the first node of a linear simple-linked list of integers, q a pointer to the last node of the same list, list that has the structure:

```
struct node
  {
    int info;
    struct node *next;
  };
typedef struct node Tnode;
typedef Tnode *ref;
ref p,q;
```

Which is the effect of the calling of the following function?

```
void func(void)
  {
    ref r;
    r=(ref)malloc(sizeof(Tnod e));
    printf("Input info:");
    scanf("%d",&(r->info));
    r->next=NULL;
    q->next=r;
    q=r;
  }
```

(a) creates a new node and inserts it at the tail of the list assigned by p , non-empty, which has the last node of address q;

(b) creates a new node and inserts it at the head of the list;

(c) modifies the content of the last node of the list, the address remaining the same;

(d) modifies the address of the last node, the content remaining unchanged;

(e) none of the actions above.

17. Let us consider the following C code:

```
  struct node
    {
       int key;
       char info[10];
       struct node *next;
    };
  typedef struct node Tnode;
  typedef Tnode *ref;
  ref p;
  /* p is a pointer to the first element of the list  */
```

The previous description can be used to describe a data structure of type:

(a) linear double linked list with integer elements;

(b) circular simple linked with integer elements;

(c) double linked circular list with integer elements;

(d) binary tree with integer elements;

(e) simple linked linear list with integer elements.

18. Let p be a pointer to the first node of a simple linked list of integers with the structure:

```
struct node
  {
    int info;
    struct node *next;
  };
typedef struct node Tnode;
typedef Tnode *ref;
ref p;
```

Which is the action of the call of the function remove?

```
void remove(ref r)
 {
  ref s;
  if (r->next==NULL)
    printf("Error:the node to be removed does not exist \n");
  else
     {
      s=r->next;
      r->next=s->next;
      free(s);
     }
 }
```

(a) removing the node after the node specified by r;

(b) removing the node specified by r;

(c) removing a different node from the last node of the list.

19. Let us consider the following C code of a double linked list structure:

```
struct node
  {
    int info;
    struct nod *ant,*next;
  };typedef struct node Tnode;typedef Tnode *ref;
ref p;
```

Which of the following statements are false:

(a) info is the information member (an integer number);

(b) the presence of the two pointers allows to iterate the list in both directions (from the first node to the last one and from the last node to the first on e);

(c) a double linked list allows to insert a new element only at one end of list;

(d) all other answers are true.

20. Let us consider the C code:

```
  struct node
     {
   int element;
   struct nod *next, *ant;
  };
 typedef struct node Tnode;
 typedef Tnode *ref;
 ref p,r;
```

Which of the following code lines should be placed in the gaps of the function insertion in order to insert a new node after the node pointed by `r` in a double linked list with the structure defined above (knowing that

```
 r->next
```

is not NULL):

```
void insertion(void)
{
  ref succ, s;
  succ = r->next;
  s=(ref)malloc(sizeof(Tnod e));
  printf("Input element=");scanf("%d", &s->element);
 .....
}
```

(a)
```
s->next = succ;
  succ->ant = s;
  s->ant = r;
  r->next = s;
```

(b)
```
s->next = succ;
  r->next = s;
  s->ant = r;
  succ->ant = s;
```

(c)
```
s->ant = r;
  s->next = succ;
  r->next = s;
  succ->ant = s;
```

21. Let us consider the following C code:

```
struct node
 {
   int element;
   struct node *next, *ant;
  };
typedef struct node Tnode;
typedef Tnode *ref;

ref p,r;
void insertion(void)
{
  ref  pred, s;
  pred = r->ant;
  s=(ref)malloc(sizeof(Tnod e));
  printf("Input element= ");scanf("%d", &s->element);
  .........
}
```

Which of the following code sequences should be placed in the insertion function so that a node is inserted before another node of address r, with r-¿ant!=NULL, in a double linked list with the node structure defined above.

(a)
```
s->next = r;
  r->next = s;
  s->ant = pred;
```

(b)
```
s->next = r;
  s->ant = pred;
  pred->next = s;
  r->ant = s;
```

(c)
```
s->ant = r->ant;
  s->next = r;
  r->next = s;
  pred->ant = s;
```

22. Let us consider the following C code:

```
struct node
 {
  int element;
  struct node *next, *ant;
  };
 typedef struct node Tnode;
 typedef Tnode *ref;
 ref p,r;
 /* p points to the first node of the list,
    r points to the node to be removed */
 void remove_node(void)
   {
    ref pred,suc;
    pred=r->ant;
    suc=r->next;
    if (r->next!=NULL) suc->ant=pred;
    if (r->ant!=NULL) pred->next=suc;
    ........
   }
```

Which of the following sequences must be placed in the `remove_node` function so that the node of address r is deleted from a double linked linear list, with the node structure defined above?

(a)
```
if (r==p) /* if the node to be removed is the first node*/
  p=p->next;
free(r);
```

(b)
```
p=NULL;
free(r);
```

(c)
```
if (r==p) /* if the node to be removed is the first node */
  p=NULL;
free(r);
```

23. A stack is:

    (a) a linear list in which the insertions are made at one end of the list and the deletions are made at the other end of the list
    (b) a linear list of LIFO (Last In First Out) type
    (c) a linear list with entry restrictions
    (d) a linear list in which the element manipulated is always the most recent one

24. Let us consider the following C code:

```
#define lg_max 100
typedef int node;
typedef struct
  {
     int top;
     nod elements[lg_max];
  } stack;
stack s;
node x;
int er;
void push(node x,stack *s)
 {
  if ((*s).top==0)
   {
    er=1;
    printf("The stack is full.\n");
   }
   else
    {
     ...
    }
 } /* push */
```

Which of the following sequences complete the function push so that it inserts an element in the stack s?

(a)
```
s.top=s.top-1;
  s.elements[s.top]=x;
```
(b)
```
(*s).top=(*s).top+1;
  (*s).elements[(*s).top]=x;
```
(c)
```
(*s).top=(*s).top-1;
  (*s).elements[(*s).top]=x;
```

25. Which of the following statements are false ?
    (a) the stack is a list in which the insertion, removal and any other access is done at a single head named top of the stack;
    (b) a stack can be implemented as static structure using an array or as dynamic structure using pointers;
    (c) a stack is a list of type FIFO (First In First Out)
    (d) the maximum numbers of elements of a stack implemented with an array is not limited within only the declaration of the array;

26. A data structure of type QUEUE is:
    (a) a structure of type array with restrictions at the entry
    (b) a linear list of type FIFO (First in First out);
    (c) a linear list in which the operations are made only at a head of the list;
    (d) a linear list in which the insertions are made at one head of the list, the removing and or what other acces is made at the other head of the list

27. Which of the following operations are typical operations for a queue implemented through a linked list:
    (a) creating
    (b) initialization
    (c) adding one element in front
    (d) adding one element in back
    (e) deleting the element from the front
    (f) deleting the element from the back
    (g) vizualization;
    (h) searching the key element of data in the queue

28. Let us consider the following C code:

```
typedef int elType;
typedef struct node
 {
  elType element;
  struct node *next;
 } Tnode;
typedef  Tnode *ref;
typedef struct
 {
  ref  front,back;
 } queue;
queue c;
int empty(queue c)
 {
  if (c.front==c.back)
    return 1;
    else  return 0;
 } /* empty */
```

The funtion empty(c) realizes:

(a) test if the queue is empty

(b) the test of the empty queue, when the queue c is implemented by using the above structure, with elements from the head of the list

(c) the test of the empty queue is implemented through the above structure without the head of the list

29. Let us consider the following C code:

```
#define Lg_max  10
typedef int elType;
typedef struct
   {
    elType elements[Lg_max];
    int front,back;
   } queue;
queue c;
int advance(int i)
     {
      if(i== Lg_max -1)
return 0;
        else return i+1;
     } /* advance */
```

The data structure defined above is:

(a) a queue implemented using a circular array;

(b) a stack implemented using an array;

(c) a double linked queue.

30. Let us consider the following C code :

```
struct node
  {
     int key;
     struct node  *left,*right;
  };
typedef struct node Tnode;
typedef Tnode * ref;
ref root;
```

In a sorted binary tree with N¿0 nodes, implemented by the above description, the maximum number of key comparisons for the searching of a given key node, is:

(a) at least equal with [log2N]+1;

(b) the smallest possible if and only if the height of the tree is minimum;

(c) the smallest possible if and only if the tree is perfectly balanced;

## 4 Graph Theory and Combinatorics

1. What is the rank of the permutation $\langle 6, 1, 3, 2, 5, 4 \rangle$ in the lexicographic enumeration of all permutations?

   (a) 411

   (b) 412

   (c) 605

   (d) 607

2. How many permutations of the letters ABCDEFGH contain the substring BCD?

   (a) 6!

   (b) 8!

   (c) 8!/3

   (d) $2^5$

3. How many strings can be produced by permuting the letter of the string SUCCESS?

   (a) 7!

   (b) $2^7$

   (c) 420

   (d) 12

4. How many integer numbers between 1 and 1000 are divisible with 7, but are not divisible with 3?

   (a) 93

   (b) 95

   (c) 92

   (d) 136

5. In a drawer there are 8 brown socks and 12 black socks. A child picks socks from the drawer at random in the dark. How many socks must he take out to be sure that he picked at least two black socks?

   (a) 3

   (b) 10

   (c) 14

   (d) 9

6. Which is the 6-permutation with repetition of the set $\{1, 2\}$ with rank 17 in the lexicographic order?
   (a) $\langle 012121 \rangle$
   (b) $\langle 010001 \rangle$
   (c) $\langle 121121 \rangle$
   (d) $\langle 121112 \rangle$

7. How many subsets with more than two elements has a set with six elements?
   (a) 32
   (b) 61
   (c) 42
   (d) 21

8. What is the general form of $a_n$ which satisfies the recurrence relation

$$a_n = -3\,a_{n-1} - 3\,a_{n-2} - a_{n-3}$$

   with initial conditions $a_0 = 1, a_1 = -2$, and $a_2 = -1$?
   (a) $a_n = 2\,n^2 - 5\,n + 1$
   (b) $a_n = -\frac{1}{2}n^3 + \frac{7}{2}n^2 - 6\,n + 1$
   (c) $a_n = -4 \cdot (-2)^n + (5n + 5)(-1)^n$
   (d) $a_n = (1 + 3\,n - 2\,n^2) \cdot (-1)^n$

9. A message sent over a communication channel is a sequence of 2 kinds of signals: type $A$ signals which last 1 microsecond, and type $B$ signals which last 2 microseconds. For example, the message $ABAAB$ lasts $3 \times 1 + 2 \times 2 = 7$ microseconds.

   Let $a_n$ be the number of different messages that last $n$ microseconds. What is the value of $a_{10}$?
   (a) 89
   (b) 55
   (c) 2917
   (d) 144

10. In the Internet, which is made up of interconnected physical networks of computers, each network connection of a computer is assigned an Internet address. In IPv4 Internet protocol, every Internet address is a 32-bit string, made of a network number (*netid*) followed by a host number (*hostid*). There are 3 types of Internet addresses:
    (a) Class A: these are of the form $0\,\underbrace{b_1b_2b_3b_4\ldots b_7}_{\text{netid}}\underbrace{b_8b_9\ldots b_{30}b_{31}}_{\text{hostid}}$

(b) Class B: these are of the form $10\,\underbrace{b_2b_3b_4\ldots b_{15}}_{\text{netid}}\,\underbrace{b_{16}b_{17}\ldots b_{30}b_{31}}_{\text{hostid}}$

(c) Class C: these are of the form $110\,\underbrace{b_3b_4b_5\ldots b_{23}}_{\text{netid}}\,\underbrace{b_{24}b_{25}\ldots b_{30}b_{31}}_{\text{hostid}}$

How many different IPv4 addresses are available for the internet network connections?

(a) $2^{29}$

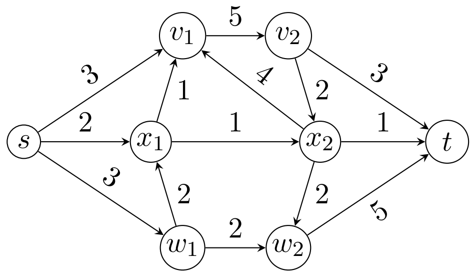
(b) $2^{30} \cdot 2^{31}$

(c) $2^{31}$

(d) $7 \cdot 2^{29}$

(e) $2^{32}$

11. Consider the flow network $G$ with source $s$ and sink (destination) $t$:



What is the value of the maximum flow in $G$?

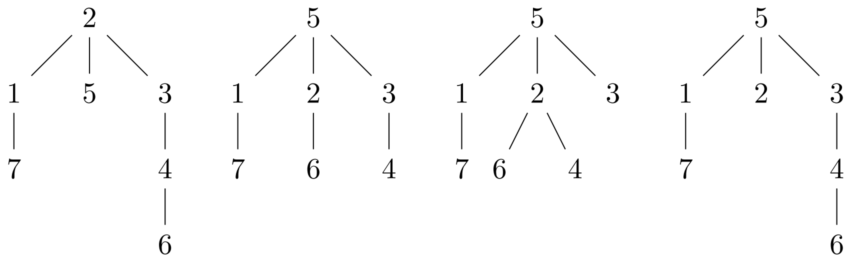
(a) 8

(b) 5

(c) 6

(d) 7

12. Which of the following trees has Prüfer sequence 5,4,3,5,1?



$T_1$ $T_2$ $T_3$ $T_4$

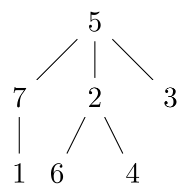
(a) $T_1$

(b) $T_2$

(c) $T_3$

(d) $T_4$

13. What is the Prüfer sequence of the following tree:



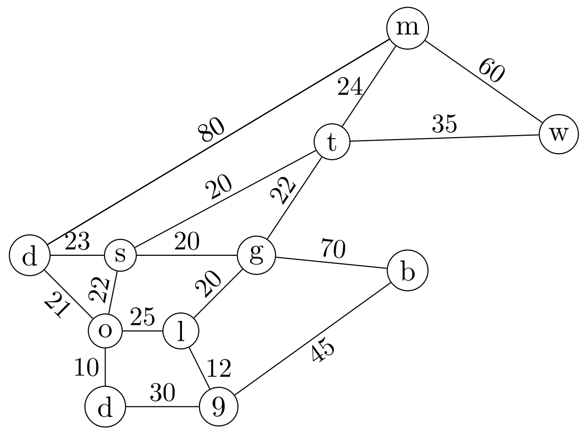
(a) $7, 5, 2, 2, 5$

(b) $1, 3, 4, 5, 2$

(c) $7, 2, 5, 5, 5$

(d) $3, 4, 6, 2, 2$

14. Consider the following weighted graph



What is the total weight of a minimum spanning tree of this graph?

(a) 229

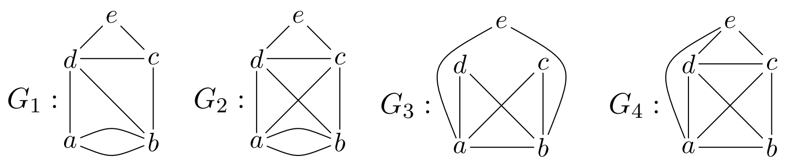
(b) 216

(c) 230

(d) 234

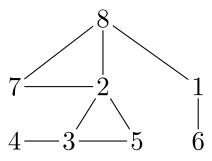
15. Which of the following is the canonical cyclic structure of the cyclic structure $(1,3,5)(7)(6,2,8)(10,4,9)$?
    (a) $(1,3,5)(7)(2,6,8)(4,9,10)$
    (b) $(1,3,5)(2,6,8)(4,10,9)(7)$
    (c) $(1,3,5)(2,8,6)(4,9,10)(7)$
    (d) $(1,3,5)(7)(2,8,6)(4,9,10)$

16. Which of the following graphs is Eulerian?



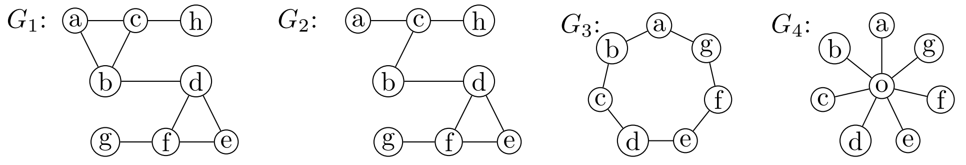
    (a) $G_1, G_4$
    (b) $G_2, G_3$
    (c) none
    (d) $G_4$
    (e) $G_1, G_3$

17. Consider the following graph:



    How many different colourings with 3 colours has this graph?
    (a) 0
    (b) 84
    (c) 120
    (d) 96
    (e) 256

18. Which of the following graphs has a perfect matching?



(a) $G_1$
(b) $G_2$
(c) $G_3$
(d) $G_4$

19. Which of the following families of sets has a system of distinct representatives?

(a) $\{1\}, \{1,2\}, \{2\}, \{3,4,5\}, \{1,2,3,4,5\}$
(b) $\{1,2\}, \{2,3\}, \{1,2,3\}, \{1,3\}, \{1,2,3,4\}, \{1,2,4,5\}$
(c) $\{1,2,3\}, \{2,3,4\}, \{3,4,5\}, \{4,5\}, \{1,2,5\}$
(d) $\{1,2,5\}, \{1,5\}, \{1,2\}, \{2,5\}, \{1,3\}, \{6\}, \{4,6,7\}$

20. How many different trees with 5 nodes, labeled with numbers from 1 to 5, exist?

(a) 10
(b) 273
(c) 32
(d) 120
(e) 125

21. Let $G$ be a graph with $n$ nodes. Indicate all statements which are equivalent with the fact that $G$ is a tree:

(1) $G$ is a connected graph without cycles.
(2) $G$ has $n-1$ edges.
(3) $G$ has no cycles, and if one edge is added to $G$, exactly one cycle is created.
(4) Every pair of nodes of $G$ is connected by one and only one simple path.

(a) $1, 2, 3, 4$
(b) $1, 2, 3$
(c) $1, 3, 4$
(d) $1$

## 5 Formal Languages and Automata Theory

1. The language generated by the grammar $G = (V_N, V_T, S, P)$, where $V_N = \{S\}$; $V_T = \{0, 1, 2\}$; $P = \{S \to 0S0|1S1|2S2|0\}$, is:

   (a) $L = \{0^n1^n2^n|n \geq 1\}$;

   (b) $L = \{w \in \{0, 1, 2\}^*|w$ palindrome centered in $0\}$;

   (c) $L = \{0^n1^m2^p|n, m, p \in N\}$;

   (d) $L = \{w0x|w, x \in \{0, 1, 2\}^*, x$ mirrored version of $w\}$;

2. The language generated by the grammar $G = (V_N, V_T, S, P)$, where $V_N = \{S\}$; $V_T = \{a, b\}$; $P = \{S \to aSb|ab\}$, is:

   (a) $L = \{a^nb^m|n, m \geq 1\}$;

   (b) $L = \{w \in \{a, b\}^*|w$ contains the infix $ab\}$;

   (c) $L = \{a^nb^n|n \geq 0\}$;

   (d) $L = \{a^nb^n|n \geq 1\}$;

3. The language generated by the grammar $G = (V_N, V_T, S, P)$, where $V_N = \{S\}$; $V_T = \{a, b\}$; $P = \{S \to bSb|bAb, A \to aA|aa\}$, is:

   (a) $L = \{b^naab^n|n \geq 1\}$;

   (b) $L = \{b^naaa^mb^n|n \geq 1, m \geq 0\}$;

   (c) $L = \{b^n(aa)^mb^n|n, m \geq 1\}$;

   (d) $L = \{b^na^mb^n|n \geq 1, m \geq 2\}$;

4. The language generated by the grammar $G = (V_N, V_T, S, P)$, where $V_N = \{S\}$; $V_T = \{a, b\}$; $P = \{S \to aSb|aAb, A \to aA|\lambda\}$, is:

   (a) $L = \{a^mb^n|m \geq n \geq 1, \}$;

   (b) $L = \{a^na^mb^n|n \geq 0, m \geq 0\}$;

   (c) $L = \{a^{i+1}b^i|i \geq 1\}$;

   (d) $L = \{a^{n+i}b^n|n \geq 1, i \geq 0\}$;

5. The language generated by the grammar $G = (V_N, V_T, S, P)$, where $V_N = \{S, A, B\}$, $V_T = \{a, b, c\}$, $P = \{S \to abc|aAbc, Ab \to bA, Ac \to Bbcc, bB \to Bb, aB \to aaA|aa\}$ , is:

   (a) $L = \{a^nb^mc^p|n, m, p \geq 1\}$;

   (b) $L = \{w \in \{a, b, c\}^*|w$ contains at least three letters $\}$;

   (c) $L = \{w \in \{a, b, c\}^*|w$ palindrome centeed in $abc\}$;

   (d) $L = \{a^nb^nc^n|n \geq 1\}$;

6. The language generated by the grammar $G = (V_N, V_T, S, P)$, where $V_N = \{S, A\}$, $V_T = \{a, b, c\}$, $P = \{S \to A|aS|bS|cS, A \to Aa|Ab|Ac|\lambda\}$, is :
   (a) $L = \{a^n b^m c^p | n, m, p \geq 0\}$;
   (b) $L = \{w \in \{a, b, c\}^* | w \text{ contains at least a letter }\}$;
   (c) $L = \{a, b, c\}^*$;
   (d) $L = \{a^n b^n c^n | n \geq 1\}$;

7. The language generated by the grammar $G = (V_N, V_T, S, P)$, where $V_N = \{S, A, B, C, D, E, F\}$, $V_T = \{a, b, \ldots, z\}$, $P = \{S \to iA|aB, A \to oC, B \to nD, C \to n, D \to a\}$, is :
   (a) $L = \{ion, ana\}$;
   (b) $L = \{ion, ani, ina, aia, iao\}$;
   (c) $L = \{w \in \{a, b, \ldots, z\}^* | w \text{ starts with } a \text{ or } i \text{ and ends with } a \text{ or } n \}$;

8. The language generated by the grammar $G = (V_N, V_T, S, P)$, unde $V_N = \{S\}$, $V_T = \{a, b\}$, $P = \{S \to aS|bS|a\}$, is:
   (a) $L = \{a^n a, b^n a | n \geq 0\}$;
   (b) $L = \{wa | w \in \{a, b\}^*\}$;
   (c) $L = \{w \in \{a, b\}^* | w \text{ ends with } a \}$;
   (d) $L = \{w \in \{a, b\}^* | w \text{ starts with } a \}$;

9. The language generated by the grammar $G = (V_N, V_T, S, P)$, unde $V_N = \{S, X\}$, $V_T = \{a, b, c\}$, $P = \{S \to aS|bS|cS|abbaX, X \to aX|bX|cX|\lambda\}$, is:
   (a) $L = \{w \in \{a, b, c\}^* | w \text{ contains } abba \}$;
   (b) $L = \{wabbaw | w \in \{a, b, c\}^*\}$;
   (c) $L = \{w \in \{a, b, c\}^* | w \text{ ends with } abba \}$;
   (d) $L = \{wabbax | w, x \in \{a, b, c\}^*\}$;
   (e) $L = \{w \in \{a, b, c\}^* | w \text{ starts with } abba \}$;

10. The rules of the grammar $G = (V_N, V_T, S, P)$, where $V_N = \{S\}$, $V_T = \{PCR, PDAR, UDMR\}$, $P = \{S \to PCR|PDAR|UDMR\}$, satisfy the constraints imposed to grammars which are:
    (a) regular (type 3);
    (b) context independent (type 2);
    (c) context dependent (type 1);
    (d) of type 0, 1, 2, 3;

11. The rules of the grammar $G = (V_N, V_T, S, P)$, unde $V_N = \{S, X\}$, $V_T = \{a, b, c\}$, $P = \{S \to aS|bS|cS|abbaX, X \to aX|bX|cX|\lambda\}$, satisfy the constraints imposed to grammars which are:

(a) regular (type 3);
(b) context independent (type 2);
(c) context dependent (type 1);
(d) of type 0, 1, 2, 3;

12. The rules of the grammar $G = (V_N, V_T, S, P)$, where $P = \{S \to abc|aAbc,\ Ab \to bA,\ Ac \to Bbcc,\ bB \to Bb,\ aB \to aaA|aa\}$, satisfy the constraints imposed to grammars which are:

    (a) regular (type 3);
    (b) context independent (type 2);
    (c) context dependent (type 1);
    (d) of type 0;

13. The rules of the grammar $G = (V_N, V_T, S, P)$, where $P = \{S \to aSb|\lambda\}$, satisfy the constraints imposed to grammars which are:

    (a) regular (type 3);
    (b) context free (type 2);
    (c) context sensitive (type 1);
    (d) computable (type 0);

14. The rules of the grammar $G = (V_N, V_T, S, P)$, unde $P = \{S \to aS|bS\lambda\}$, satisfy the constraints imposed to grammars which are:

    (a) regular (type 3);
    (b) context independent (type 2);
    (c) context dependent (type 1);
    (d) of type 0;

15. The regular expression which denotes the language $L = \{w|$ strings of 0 and 1 which end in 1 $\}$, is:

    (a) $(0|1)^*1$;
    (b) $1|(0|1)^*$;;
    (c) $(0^*|1^*)^*1$;
    (d) $(1|0|1)^*1$;

16. The regular expression which denotes the language $L = \{w|$ strings of 0 and 1 which contain at least one symbol 1 $\}$, is:

    (a) $(0|1)^*1$;
    (b) $1|(0|1)^*1(1|0)^*$;

(c) $0^*11^*$;

(d) $(0|1)^*1(0|1)^*$;

17. The regular expression which denotes the language $L = \{w|$ strings of 0 and 1 which contain at least one symbol $\}$ is:

(a) $(0|1)^*1$;

(b) $1|0|(0|1)^*$;

(c) $(0^*|1^*)^*|1|0$;

(d) $(1|0)(0|1)^*$;

18. The regular expression which denotes the language $L = \{ana, ani, ina, ini\}$ is:

(a) $ana|ani|ina|ini$;

(b) $(a|i)n(a|i)$;

(c) $a^*ni^*$;

(d) $(a|i)^*n(a|i)^*$;

19. The regular expressions denote:

(a) regular languages;

(b) languages of type 0 and 3;

(c) languages recognized by finite deterministic automata;

(d) only finite languages;

20. The language $L$ denoted by the regular expression $(a|i)n(a|i)$ is:

(a) $L = \{ana, ani, ina, ini\}$;

(b) $L = \{w \in \{a, n, i\}^*|w$ starts and ends with $a$ or $i\}$;

(c) $L = \{w \in \{a, n, i\}^*|w$ starts with $a$ or $i$ and ends with $na$ or $ni$ $\}$;

(d) $L = \{w \in \{a, n, i\}^*|w$ starts and ends with $a$ or $i$, and letter $n$ is in the middle of word $w$ $\}$;

21. The language $L$ denoted by the regular expression $(0|1)(0|1)^*$ is:

(a) $L = \{0, 1, 00, 01, 10, 11, 000, \ldots\}$;

(b) $L = \{w \in \{0, 1\}^*|w$ starts with 0 or 1$\}$;

(c) $L = \{0, 1\}^*$;

22. The language $L$ denoted by the regular expression $01^*|1$; is:

(a) $L = \{0, 1, 00, 01, 10, 11, 000, \ldots\}$;

(b) $L = \{w \in \{0, 1\}^*|w$ starts with 1 $\}$;

(c) $L = \{w \in \{0,1\}^* | w \text{ starts with 1 }\}$;

(d) $L = \{01^n | n \geq 1\} \bigcup \{1\}$;

23. The language $L$ denoted by the regular expression $(11)^*1$ is:

(a) $L = \{1, 11, 111, 1111, \ldots\}$;

(b) $L = \{w \in \{0,1\}^* | w \text{ has an odd number of occurences of symbol } 1\}$;

(c) $L = \{1w | w = 1^{2n}, n \in N\}$;

(d) $L = \{1^{2n+1}, n \in N\}$;

24. The language $L$ denoted by the regular expression $(1|0)^*0(0|1)$ is:

(a) $L = \{0, 1, 00, 01, 10, 11, \ldots\}$;

(b) $L = \{w \in \{0,1\}^* | w \text{ ends with 00 or 01 }\}$;

(c) $L = \{w \in \{0,1\}^* | w \text{ has at least one symbol 0 }\}$;

(d) $L = \{w \in \{0,1\}^* | w \text{ has 0 on the position before the last one }\}$;

## 6 Answers

### Algorithms

1. 1c
2. 2d
3. 3d
4. 4c,4d
5. 5a,5c
6. 6c
7. 7c
8. 8b
9. 9a,9d
10. 10d
11. 11b,11e
12. 12d
13. 13b
14. 14c,14d,14e
15. 15b
16. 16b,16c
17. 17a,17b
18. 18c
19. 19d
20. 20c
21. 21b
22. 22b,22c
23. 23b
24. 24b,24d
25. 25a,25b,25d
26. 26b,26e
27. 27d,27f
28. 28d
29. 29b,29d,29f
30. 30b

### Logic for Computer Science

1. 1c,1d
2. 2a,2c
3. 3a,3c,3d
4. 4a,4b
5. 5b,5c
6. 6a,6c
7. 7a,7b,7c
8. 8b
9. 9a,9b,9c,9d
10. 10c,10d
11. 11b
12. 12c,12d
13. 13a,13c,13d
14. 14b,14c
15. 15b
16. 16a
17. 17b
18. 18b,18c
19. 19c
20. 20a

## Data Structures

1. 1b
2. 2b, 2c, 2e
3. 3a,3c,3d,3e
4. 4b
5. 5c
6. 6a,6c
7. 7b,7d
8. 8b
9. 9d
10. 10b
11. 11d
12. 12a
13. 13c
14. 14c
15. 15a
16. 16a
17. 17b, 17e
18. 18a
19. 19c,19d
20. 20a, 20b, 20c
21. 21b
22. 22a
23. 23b, 23d
24. 24c
25. 25c, 25d
26. 26b, 26d
27. 27d, 27e
28. 28b
29. 29a
30. 30b, 30c

## Graph Theory and Combinatorics

1. 1d
2. 2a
3. 3c
4. 4b
5. 5b
6. 6d
7. 7c
8. 8d
9. 9a
10. 10d
11. 11d
12. 12d
13. 13a
14. 14a
15. 15c
16. 16b
17. 17d
18. 18a
19. 19c
20. 20e
21. 21c

## Formal Languages and Automata Theory

1. 1b,1d
2. 2d
3. 3b,3d
4. 4a,4d
5. 5d
6. 6c
7. 7a
8. 8b,8c
9. 9a,9d
10. 10a,10b,10c,10d
11. 11b
12. 12d
13. 13b,13d
14. 14a,14b,14d
15. 15a,15c,15d
16. 16b,16d
17. 17d
18. 18a,18b
19. 19a,19c
20. 20a
21. 21a,21b
22. 22d
23. 23c,23d
24. 24b,24d