ПРОГРАМИРАНЕ 1

Диана Иванова

СЪДЪРЖАНИЕ:

БРОЙНИ СИСТЕМИ

1. ОСНОВНИ БРОЙНИ СИСТЕМИ

1.1. ОБЩИ ПОЛОЖЕНИЯ

Въпреки, че съществуват много бройни системи NUMBER SYSTEMS, ние ще се занимаваме с четири. Те са:
Десетична система, която ползува за основа числото 10;
Двоична система, която ползува за основа числото 2;
Осмична система, която ползува за основа числото 8;
Шестнайсетична система, която ползува за основа числото 16;

Тези системи са широко използувани в цифровите системи и компютрите.

Ние ще разглеждаме двоичните, осмичните и шестнайсетичните бройни системи като ги сравним с нашата много добре позната десетична система.

1.2. ДЕСЕТИЧНА БРОЙНА СИСТЕМА.

В десетичната бройна система числата са представени чрез цифрите от 0 до 9 и чрез даване на стойност на цифровата позиция. Така десетичното цяло число 537 е получено от: $5x10^2$ + 3x10' + 7x10°
Различните цифрови позиции на ляво в десетичното число са с нарастващи степени на 10.

Хиляди	Стотици	Десетици	Единици
10^3	10^2	10^1	10^0

Както знаем дадено число може да се представи с десетични знаци. За дробни числа прилагаме подобна стойност (приложение), цифри зад десетична точка са записани в отрицателна степен на 10.

ПРИМЕР: $2786{,}134_{10}$ може да се смята като:

$2x10^3 + 7x10^2 + 8x10^1 + 6x10^0 + 1x10^{-1} + 3x10^{-2} + 4x10^{-3}$

Десетичната система е казано да има основа или база 10, защото тя използува 10 цифри и съседни цифрови позиции различаващи се на една степен на 10.

За едно число знакът използуван за долен индекс, който показва каква е основата (базата) се записва по следния начин:

123_{10} означава десетичното число 123;

11011_2 означава двоично число 11011;

567_8 означава осмичното число 567;

1.3. ДВОИЧНА БРОЙНА СИСТЕМА.

Двоичната система използува само две цифри 0 и 1, но точно същите основни принципи се прилагат както за десетичната бройна система. Всяка съседна цифрова позиция (бит) се различава със степен на 2. Така двоичното число 10110_2 може да бъде написано като:

2^4	2^3	2^2	2^1	2^0
16	8	4	2	1
1	0	1	1	0

или като $1x2^4 + 0x2^3 + 1x2^2 + 1x2^1 + 0x2^0$

Десетичният еквивалент на двоичното число лесно се получава чрез сумиране заедно различните степени на две за числото.

$1x2^4 + 0x2^3 + 1x2^2 + 1x2^1 + 0x2^0 = 16 + 0 + 4 + 2 + 0 = 22_{10}$

Опитайте един или два примера самостоятелно:

ЗАДАЧИ: Преобразувайте следните двоични числа в техния десетичен еквивалент.

а) 101_2 Отг. 5_{10}

б) 1111_2 Отг. 15_{10}

в) 100111_2 Отг. 39_{10}

1.4. ОСМИЧНА БРОЙНА СИСТЕМА.

Тази система използува осем за основа или база.така че всяка цифрова позиция е степен на 8.

8^3	8^2	8^1	8^0	8^{-1}	8^{-2}	8^{-3}
512	64	8	1	1/8	1/64	1/512

ПРИМЕР: Така осмичното число 56_8 може да бъде преобразувано в десетично като:

$56_8 = 5x8^1 + 6x8^0 = 5x8 + 6\ 8^0 = 40 + 6 = 46_{10}$

ПРИМЕР: Аналогично и за осмичното 777_8 десетичният еквивалент е:

$777_8 = 7x8^2 + 7x8^1 + 7x8^0 = 7x64 + 7x8 + 7 = 448 + 56 + 7 = 511_{10}$ '

1.5. ШЕСТНАЙСЕТИЧНА БРОЙНА СИСТЕМА.

Тази система използува шестнадесет за основа или база, така че всяка цифрова позиция е степен на 16. Тъй като тази основа използува повече от 10 символа, първите 6 букви от английската азбуката се използуват да представят числата от 10 до 15.

10 = A; 13= D;
11 = B; 14 = E;
12 = C; 15= F.

Таблицата по-долу показва десетичните, двоични, осмични и шестнадесетични еквиваленти на първите 21 десетични числа.

Десетични Двоични Осмични Шестнадесетични

0	00000	00	0
1	00001	01	1
2	00010	02	2
3	00011	03	3
4	00100	04	4
5	00101	05	5
6	00110	06	6
7	00111	07	7
8	01000	10	8
9	01001	11	9
10	01010	12	A
11	01011	13	B
12	01100	14	C
13	01101	15	D
14	01110	16	E
15	01111	17	F
16	10000	20	10
17	10001	21	11
18	10010	22	12
19	10011	23	13
20	10100	24	14

Десетични еквиваленти от шестнадесетични числа получаваме аналогично

$567_{16} = 5x16^2 + 6x16^1 + 7x16^0 = 5x256 + 6x16 + 7 = 1280 + 96 + 7 = 1383_{10}$

$FAB4_{16} = Fx16^3 + Ax16^2 + Bx16^1 + 4x16^0 = 15x16^3 + 10x16^2 + 11x16^1 + 4x16^0 = 15x4096 + 10x256 + 11x16 + 4x1 = 61440 + 2560 + 176 + 4 = 64180_{10}$

ЗАДАЧА: Преобразувайте шестнадесетичното число B65F в десетично самостоятелно

Отг. 46687_{10}

Трябва да отбележим, че двоичната система изисква около три пъти повече цифри, за да дефинира числото в сравнение с другите (осмична и шестнадесетична), като използува два символа 0 и 1. Тя се използува вътре в цифровия компютър, а другите две при обмена човек компютър.

2.ОСНОВНИ ПРЕОБРАЗУВАНИЯ

Ние вече видяхме как да преобразуваме двоична, осмична и шеснадесетична система в десетична.

Сега ще разгледаме как да преобразуваме десетично число в двоично.

2.1. ПРЕОБРАЗУВАНЕ НА ДЕСЕТИЧНО ЧИСЛО В ДВОИЧНО

2.1.1. Цяло десетично число

Десетично в двоично преобразуване се прави чрез повтарящо се деление на 2. Това е демонстрирано със следния пример.

ПРИМЕР: Преобразувай десетичното число 182 в двоично

Числото се дели на 2 и остатъкът ако има такъв се записва успоредно. Резултатът от делението (частното) се записва отдолу и процесът продължава, докато цялата част на частното стане равна на 0. Двоичното число се намира чрез получената редица от остатъците, като то е с най-младшия значещ бит (LSB) на върха на редицата от получените остатъци от делението с 2.

182:2	остатък	0
91:2	остатък	1
45:2	остатък	1
22:2	остатък	0
11:2	остатък	1
5:2	остатък	1
2:2	остатък	0
1:2	остатък	1

Така $182_{10} = 10110110_2$

За всяко преобразуване една валидна проверка бе се осъществила веднага. И така използувайки преобразуването на двоичното число в десетично ние имаме:

$$1x2^7 + 0x2^6 + 1x2^5 + 1x2^4 + 0x2^3 + 1x2^2 + 1x2^1 + 0x2^0 =$$

$$= 128 + 0 + 32 + 16 + 0 + 4 + 2 + 0 = 182_{10}$$

ЗАДАЧА: Опитай самостоятелно преобразуването на десетично 93 в двоично и провери твоя резултат.
отг. 1011101_2

2.2. ПРЕОБРАЗУВАНЕ НА ДВОИЧНО ЧИСЛО В ОСМИЧНО И ОБРАТНО

2.2.1. Преобразуване на цяло двоично число в осмично

Двоичното число се разделя на групи от три бита започвайки от младшия разряд до края на числото и се записва съответният осмичен еквивалент на групите от три бита.

ПРИМЕР: 11101011100111_2 = 11 101 011 100 111_2 = 35347_8

3 5 3 4 7

ЗАДАЧА: Преобразувай двоичното число 1101010110111_2 в осмично. Отг. 15267_8

2.2.2. Преобразуване на цяло осмично число в двоично

Този процес е обратен на описания в точка 2.2.1. Всяко осмично число се заменя с неговия три битов двоичен еквивалент. Така, че числото

$12345_8=$ 1 2 3 4 $5_8=1010011100101_2$

001 010 011 100 101

2.2.3. Преобразуване на дробно двоично число в осмично

Тази процедура е много сходна на тази за цели числа (2.2.1). Двоичното дробно число се разделя на групи от три бита започвайки от двоичната точка и движещ на дясно. Всяка група от три бита се преобразува в еквивалентен осмичен знак.

ПРИМЕР:

Преобразувай $0{,}1101010_2$ в осмично и провери преобразуването чрез замяна на двете осмично и двоично числа в десетично.

Групирайки двоичното число в тройки от десетичната точка на дясно ние имаме 0,110.101 и тогава заменяйки групите от три бита с осмичния еквивалент ние получаваме

0,110 101 $=0{,}65_8$ и така $0{,}1101010_2 = 0{,}65_8$

6 5

ПРОВЕРКА: $0{,}1101010_2 = 0{,}5 + 0{,}25 + 0{,}0625 + 0{,}015625 = 0{,}828125_{10}$

$0{,}65_8 = 6x1/8 + 5x1/64 = 6x0{,}125 + 5x\ 0{,}015625 = 0{,}828125_{10}$

ЗАДАЧА:

Преобразувай $0{,}01010111_2$ в осмично и провери твоя резултат.

Отг. $0{,}256_8$ и $0{,}33984375_{10}$

2.3. ПРЕОБРАЗУВАНЕ НА ДВОИЧНО ЧИСЛО В ШЕСТНАДЕСЕТИЧНО И ОБРАТНО

2.3.1. Преобразуване на цяло двоично число в шестнадесетично

Този процес е аналогичен на преобразуването от двоично число в осмично (точка 2.2.1). Двоичното число се разделя на групи от четири бита започвайки от младшия разряд до края на числото и се записва съответният шестнадесетичен еквивалент на групите от четири бита.

ПРИМЕР: Преобразувай 11101011100111_2 в шестнадесетично. Раздели числото в групи започвайки от младшия разряд на ляво до края.

$= 11\ 1010\ 1110\ 0111_2$ (двоична в групи от четири)

3 10 13 7 (десетична)

3 A E 7 (шестнадесетична)

Така $11101011100111_2 = 3AE7_{16}$

ЗАДАЧА: Преобразувай двоичното число 111110110111111_2 в шестнадесетично.

Отг. $7DBF_{16}$

2.3.2. Преобразуване на цяло шестнадесетично число в двоично

Отново този процес е обратен на описания в точка 2.3.1. Всяко шестнадесетично число се заменя с неговия четири битов двоичен еквивалент.Така.че числото

$E3A5_{16}$= E 3 A 5_{16} =1110001110100101_2

1110 0011 1010 0101

ЗАДАЧА: Преобразувай шестнадесетично число $6CAE_{16}$ в двоично

Отг. 110110010101110_2

2.3.3. Преобразуване на дробно двоично число в шестнадесетично

Тази процедура е логическо продължение на метода използуван за цели числа (2.3.1). Двоичното дробно число се разделя на групи от четири бита започвайки от двоичната точка и движещ на дясно. Всяка група от четири бита се преобразува в еквивалентен шестнадесетичен знак.

ПРИМЕР: Преобразувай двоичното число $0{,}010101110_2$ в шестнадесетично

Групирайки двоичното число в четворки от десетичната точка на дясно ние имаме 0,0101 0111 0000 и тогава заменяйки групите от четири бита с шестнадесетичен еквивалент ние получаваме $0{,}570_{16}$ и така $0{,}010101110_2 = 0{,}57_{16}$

ЗАДАЧА: Преобразувай двоичното число по-долу в шестнадесетично.

$11111011010{,}11111111000111_2$ Отг. $7DA{,}FF1C_{16}$

С тези знания преобразуванията между двоична, десетична, осмична и шестнадесетична системи са лесни да се направят.

3. АРИТМЕТИЧНИ ОПЕРАЦИИ

По принцип тези операции се извършват по същият начин, както обичайната десетична аритметика. По-особено е само формирането на преноса към по-старшия разряд при събиране и заемането от по-старши разряд при изваждане.

3.1 СЪБИРАНЕ

Числата се събират поразредно, като в резултат на събирането на i-тите разреди на двете събираеми се получава i-тият разред на сумата им. 3.1.1. Двоично събиране

Двоичната аритметика е бърза и лесна тъй като събиранията могат да бъдат само четири вида:

0+0=0

1+0=1

0+1=1

1+1=0 пренос (carry) 1

Сума (Sum)

Забележете връзката между тази таблица за събиране и таблицата на истинност за елемента Изключващо ИЛИ (EXOR).

Тъй като имаме само две цифри 0 и 1, пренос се случва твърде често. ПРИМЕР: Да се съберат трите двоични числа и да се направи проверка чрез десетично събиране.

11011	27
+10011	+19
11000	24
1000110_2	70_{10}

3.1.2. Осмично събиране

Тук трябва да се помни, че пренос 1 към по-старши разряд се подава, когато сумата е по-голяма или равна на 8.

ПРИМЕР: Да се съберат двете осмични числа и да се направи проверка, чрез десетично събиране.

71_8	57_{10}
$+25_8$	$+21_{10}$
116_8	78_{10}

ЗАДАЧА: Както обикновено опитай самостоятелно следното

Събери осмичното число 625 с осмичното число 773 и провери твоя резултат. Отг.1620_8, което е еквивалентно на 912_{10}

3.1.3**.** Шестнадесетично събиране

Тук трябва да се помни, че пренос 1 към по-старши разряд се подава, когато сумата е по-голяма или равна на 16.

ПРИМЕР: Да се съберат двете шестнадесетични числа и да се направи проверка чрез десетично събиране.

$BC67_{16}$	48231_{10}
$+594A_{16}$	$+22858_{10}$
$115B1_{16}$	71089_{10}

Шестнадесетично събиране извършваме започвайки от най-младшия значещ разряд на двете числа, в случая 7 плюс A = 7 +10 = 17 , което е 16 +1, следователно сумата , която ще запишем като резултат е 1 и пренос също 1 ще добавим към следващата колона. В тази колона ние имаме 6 + 4 +1 (пренос) = 11 = B, следователно сумата , която ще запишем като резултат е B и няма пренос. В следващата колона имаме C + 9 = 12 + 9 = 21 = 16 + 5, следователно записваме 5 за сума и пренос 1. В финалната колона ние имаме B + 5 + 1 (пренос) = 11 + 5 + 1 = 17 = 16 + 1, следователно записваме 1 за сума и пренос 1. И така резултатната сума е$115B1_{16}$

ЗАДАЧА: Както обикновено опитай самостоятелно следното

Събери шестнадесетичното число ABC с шестнадесетичното число 789 и провери твоя резултат.

Отг. 1245_{16}

3.2. УМНОЖЕНИЕ

За умножение на две числа в произволна позиционна система се прилагат правилата за умножение при десетичната система, като за улеснение се използува таблицата за умножение в съответната система. 3.2.1. Умножение на цели двоични числа

Умножителната двоична таблица не би поставяла проблеми, тъй като тя заема само четири реда:

0 . 0 = 0 ; 0 . 1 = 0 ; 1 . 0 = 0; 1 . 1 = 1

Забележете връзката между тази таблица за умножение и Булевата функция И.

За умножение на числа в двоична система може да се използува едно от следните правила:

1) Умножаването на двете числа се започва от най-младшия разред на множителя. Ако този разред е 1, записва се множимото, а ако е 0, се записва един ред от нули. При 1 в следващия разред на множителя се преписва множимото, изместено с един разред вляво, а при 0 се записва ред нули, също изместени с един разред вляво. След това се преминава към следващия разред и т. н.

2) Умножаването на двете числа се започва от най-старшия разред на множителя. Ако този разред е 1, записва се множимото, а ако е 0, се записва един ред от нули. При 1 в следващия разред на множителя се преписва множимото, изместено с един разред вдясно, а при 0 се записва ред нули, също изместени с един разред вдясно. След това се преминава към следващия разред и т. н.

ПРИМЕР: Извърши умножението на двете двоични числа

1010 1
x
11010
101010000
1010100 0
101010
1000100010

Десетичният еквивалент е 21 x 26 = 546

.

ЗАДАЧИ: Бройни Системи

1. В каква БС 2*2=100?
2. В каква БС 2 * 2 = 10?
3. В каква БС 2*2=11?
4. В каква БС 4 * 4 = 31?
5. В каква БС 3 * 3 = 10?
6. Вярно ли е равенството 7 + 8 = 16 ?
7. В каква БС 71 – 36 = 33?

8. В коя бройна система $21 + 24 = 100$?
9. В коя бройна система $20 + 25 = 100$?
10. В коя бройна система $22 + 44 = 110$?
11. Ако $4 * 4 = 20$, то на какво е равно $5 * 5$ (в същата бройна система).
12. Подредете по възходящ ред: 1001, 111, 010, 100, 1101, 10001.
13. Аз завърших университета на 44 години.След една година станах 100-годишен млад човек и се ожених за 34-годишна девойка. Незначителната разлика във възрастта ни – всичко -11 години помага за това да имаме общи мечти и интереси. След немного години аз вече имах едно малко семейство от 10 деца. С какво се обяснява странното противоречие? Възстановете истинския смисъл на числата.
14. ***«1101 ноября этого года в маленьком городке Тиб, по улице 101 - Авеню в доме 111 квартира 10101 было совершено громкое преступление. У юного 1001-летнего художника были украдены 11 картин. Пострадавший оценил стоимость похищенного в 100100 рублей, сюда вошли стоимость за краски, цветные карандаши, и альбом.» Възстановете истинските данни***

15. Дадено е $a=D7_{16}$, $b=331_8$. Кое число c, записано в двоична бройна система, отговаря на условието $a<c<b$?

1) 11011001 2) 11011100 3) 11010111 4) 11011000

16. На какво е равна сумата от числата 43_8 и 56_{16}?

1) 121_8 2) 171_8 3) 69_{16} 4) 1000001_2

17. Десетичното число 59 е еквивалентно на числото 214 в дадена бройна система. Намерете основата на тази система.
18. В каква БС 33_{10} ще се запише като 53?
19. В каква БС 23_{10} ще се запише като 212?
20. В каква БС 42_{10} ще се запише като 52?
21. В каква БС $71 – 36 = 33$?
22. Запишете в троична БС текущата година.
23. $111110_2+10010_2+101_2=$
24. $10101_2+10110_2+111_2$
25. $265_8+765_8=$
26. $66666_8+6666_8=$
27. $FAD_{16}+65_{16}+CD_{16}=$
28. $9999_{16}+356D_{16}=$
29. $111110_2-10010_2-101_2$
30. $10101_2-10110_2-111_2$
31. $1265_8-765_8=$
32. $54321_8-666_8=$
33. $FAD_{16}-CD_{16}=$
34. $99999_{16}-356D_{16}=$
35. $111010_2+10110_2+110_2=$

36. $10111_2+11111_2+100_2=$
37. $565_8+777_8=$
38. $7777_8+6666_8=$
39. $F6AD_{16}+85_{16}+C1D_{16}=$
40. $AAA_{16}+356D_{16}=$
41. $111010_2-10110_2-110_2=$
42. $10111_2-11111_2-100_2=$
43. $4565_8-777_8=$
44. $7777_8-6666_8=$
45. $F6AD_{16}-C1D_{16}=$
46. $FAAA_{16}-356D_{16}=$
47. Запишете първите 20 цели числа в десетична, двоична, троична, петична и осмична бройни системи
48. Кои цели числа следват след числата:

а) 1_2;	е) 1_8;	л) F_{16};
б) 101_2;	ж) 7_8;	м) $1F_{16}$;
в) 111_2;	з) 37_8;	н) FF_{16};
г) 1111_2;	и) 177_8;	о) $9AF9_{16}$;
д) 101011_2;	к) 7777_8;	п) $CDEF_{16}$?

49. Кои цели числа предшестват числата:

а) 10_2;	е) 10_8;	л) 10_{16};
б) 1010_2;	ж) 20_8;	м)20_{16};
в) 1000_2;	з) 100_8;	н) 100_{16};
г) 10000_2;	и) 110_8;	о) $A10_{16}$;
д) 10100_2;	к) 1000_8;	п) 1000_{16} ?

50. На коя цифра завършва четното двоично число? На коя цифра завършва нечетното двоично число? На кои цифри може да завърши четното троично число?
51. Кое най-голямо десетично число може да се запише с три цифри:

а) в двоична система;
б) в осмична система;
в) в шестнадесетична система?

52. Превърнете числата в десетична система, а след това проверете резултата:

а) 1011011_2;	е) 517_8;	л) $1F_{16}$;
б) 10110111_2;	ж) 1010_8;	м) ABC_{16};
в) 011100001_2;	з) 1234_8;	н) 1010_{16};
г) $0,1000110_2$;	и) $0,34_8$;	о) $0,A4_{16}$;

д) $110100{,}11_2$; к) $123{,}41_8$; п) $1DE{,}C8_{16}$.

53. Превърнете числата от десетична система в двоична, осмична и шестнадесетична, а след това проверете резултата:

а) 125_{10}; б) 229_{10}; в) 88_{10}; г) $37{,}25_{10}$; д) $206{,}125_{10}$.

54. Превърнете числата от двоична система в осмична и шестнадесетична, а след това проверете резултата:

а) $1001111110111{,}0111_2$; г) $1011110011100{,}11_2$;
б) $1110101011{,}1011101_2$; д) $10111{,}1111101111_2$;
в) $10111001{,}101100111_2$; е) $1100010101{,}11001_2$.

Логически функции

За описание на функционирането на логическите схеми ще разгледаме някои базови понятия от алгебрата на логиката. Основите на този математически апарат са разработени от ирландския математик Бул (1815 – 1864), поради което често алгебрата на логиката се нарича Булева алгебра.

Логически функции са зависимости от вида:

(1) $Y = f (X_1, X_2, X_3, \dots Xm)$,

в които както Y, така и аргументите (променливите) X_1, X_2, X_3, $\dots X_m$ могат да приемат само две стойности: 0 и 1.

Логическите функции се задават (дефинират) по няколко начина:

а) словесно (чрез текст);

б) таблично (чрез таблици на истинност);

в) аналитично (чрез формула $Y = f (X_1, X_2, X_3, \dots X_m)$);

г) графично (чрез m-мерен куб или карти на Вейч (Карно)).

Един от начините за задаване на логически функции е чрез таблица на истинност. В нея е посочена стойността на функцията за всяко съчетание (набор) от стойности на аргументите.

Логическите функции от един аргумент имат условно обозначение и наименование дадени в таблица 1:

Таблица 1

X	0	1	Означение	Название	Логическа схема
F_1	0	0	$Y = 0$	Константа 0	-
F_2	0	1	$Y = X$	Променлива X	Повторител; схема “да”
F_3	1	0	$Y = \overline{X}$	Инверсия на X	Инвертор; схема “не”
F_4	1	1	$Y = 1$	Константа 1	-

Съществена е функцията $Y = F_3 (X) = \overline{X}$, наречена логическо отрицание или инверсия. Означава се с чертичка над аргумента и се чете “не X”. Както се вижда от таблицата, стойността на функцията е винаги обратна на стойността на аргумента.

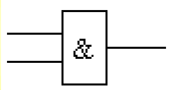
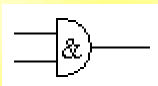
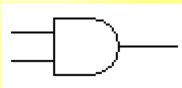
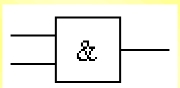
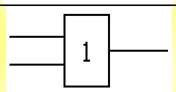
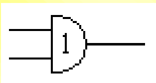
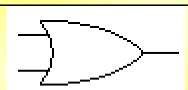
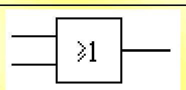
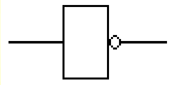
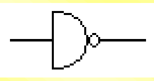
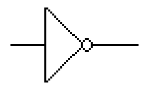
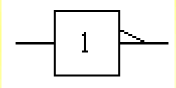
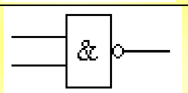
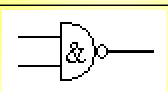
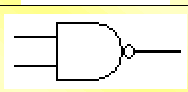
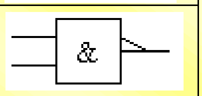
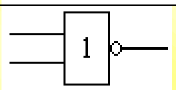
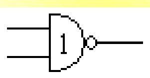
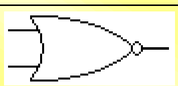
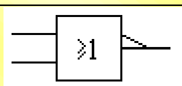
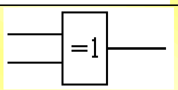
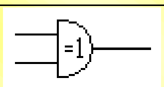
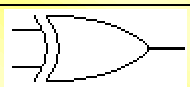
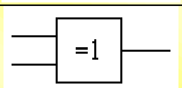
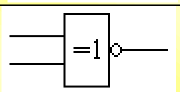
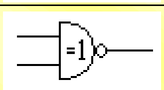
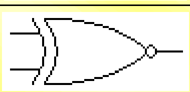
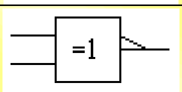
При n = 2, т.е. за логическа функция от два аргумента X_1 и X_2, съществуват $2^4 = 16$ различни логически функции. От всичките шестнадесет функции на две променливи интерес представляват функциите, показани в таблица 2:

Таблица 2

X_1	0	0	1	1	Означение	Название	Логическа схема
X_2	0	1	0	1			
F_1	0	0	0	1	$Y = X_1.X_2$	Конюнкция	И
F_6	0	1	1	0	$Y = X_1 \oplus X_2$	Сума по модул 2	Изключващо ИЛИ
F_7	0	1	1	1	$Y = X_1 \cup X_2$	Дизюнкция	ИЛИ
F_8	1	0	0	0	$Y = \overline{X_1 \cup X_2}$	Функция на Пирс	НЕ – ИЛИ
F_9	1	0	0	1	$Y = \overline{X_1 \oplus X_2}$	Логическа равнозначност	Изключващо ИЛИ – НЕ
F_{14}	1	1	1	0	$Y = \overline{X_1.X_2}$	Функция на Шефер	НЕ - И

Логическите функции дават връзката между входните променливи и изходните сигнали. Схемната реализация на функцията представлява логическа схема. В следващата таблица са показани всички общоупотребявани обозначения на логическите функции, които се използват в европейската и американската специализирана литература:

Таблица 3

Име на функцията	Английски стандарт	Стар Английски стандарт	Американски стандарт	ANSI / IEEE стандарт (91 – 1984)
И				
ИЛИ				
НЕ				
И – НЕ				
ИЛИ – НЕ				
ИЗКЛЮЧВАЩО ИЛИ				
ИЗКЛЮЧВАЩО ИЛИ - НЕ				

Теорема на Бул - ***Всяка булева функция може да се представи със съждителна формула (с помощта на конюнкция, дизюнкция и отрицание)***

1. Най-съществените причини за опростяване на една логическа функция са:
   - За реализирането на една функция трябва да се използват колкото е възможно по-малко елементи. Това *намалява цената на схемата, повишава надеждността ѝ и намалява габаритните ѝ размери.*

   - *Повишаване на бързодействието* на дадената схема. През всяка реална схема сигналът преминава за определено време. Когато сигналът преминава през много логически елементи, свързани последователно, времената на закъсненията им се сумират и се увеличава общото време за предаване на сигнала от входа към изхода.

1. БУЛЕВИ ЗАКОНИ

1.1 ОСНОВНИ ЗАВИСИМОСТИ С ЕДНА ПРОМЕНЛИВА В БУЛЕВАТА АЛГЕБРА.

Получават се чрез трите базови логически функции НЕ, ИЛИ, И, приложени за една единствена променлива.

НЕ Отрицанието на НЕ $X_1 = \overline{\overline{X_1}} = X_1$

ИЛИ Една променлива може да участвува в операция ИЛИ с константа 0, константа 1, самата себе си или своята инверсия.

$X_1 \vee 0 = X_1;\ X_1 \vee X_1 = X_1;\ X_1 \vee 1 = 1;\ X_1 \vee \overline{X_1} = 1$

И Една променлива може да участвува в операция И с константа 0, константа 1, самата себе си или своята инверсия.

$X_1.0 = 0;\ X_1.1 = X_1;\ X_1.X_1 = X_1;\ X_1.\overline{X_1} = 0$

Всички тези зависимости могат да бъдат доказани чрез заместване на двете стойности за X, във всяко уравнение и потвърждават, че то остава валидно и в двата случая.

ПРИМЕРИ: Да се докаже, че:

	а) $X_1 \vee 0 = X_1$;	б) $\overline{X_1}\, X_1 = 0$
За $X_1 = 0$,	$0 \vee 0 = 0$;	$0\ 1 = 0$;
За $X_1 = 1$,	$1 \vee 0 = 1$;	$1\ 0 = 0$.

1.2. ОСНОВНИ БУЛЕВИ ЗАКОНИ

Има три основни алгебрични закона, които се отнасят за начина по който уравненията са написани. Всеки от тях е изразен в две (двойнствени) форми – дизюнктивна (V или +) и конюнктивна (.).

а) Комутативен закон;

Този закон гласи, че нареждането на променливите не е важно. То може да се обобщи за произволен брой променливи.

(1-2) $X_1 \vee X_2 = X_2 \vee X_1$, $X_1 X_2 = X_2 X_1$;

Следователно подреждането на сигналите към входовете на логическия елемент е произволно. Всеки два сигнала могат да бъдат разменени без да се промени логическата функция реализирана от елемента.

б) Асоциативен закон;

Този закон гласи, че групирането на произволен брой променливи не променя крайния резултат. Следователно, редът в който ИЛИ операциите се изпълняват от логическите елементи е условен. Същото е вярно за И операциите.

(1-3) $(X_1 \vee X_2) \vee X_3 = X_1 \vee (X_2 \vee X_3) = X_1 \vee X_2 \vee X_3,$

$(X_1 X_2) X_3 = X_1 (X_2 X_3) = X_1 X_2 X_3$

в) Дистрибутивен закон;

Този закон гласи, че една операция приложена върху израз заграден със скоби, се разпределя върху всички променливи в скобите.

(1-4) $X_1 (X_2 \vee X_3) = (X_1X_2) \vee (X_1X_3) = X_1X_2 \vee X_1X_3$

$X_1 \vee (X_2X_3) = (X_1 \vee X_2)(X_1 \vee X_3)$

(Забележете, че втората форма на дистрибутивния закон е единствената форма, която се отличава от обикновената алгебра!)

г) Закони(Теореми) на Де Морган;

Законите на Де Морган са полезни особено за опростяване на Булеви изрази съдържащи отрицания на членове (променливи) или групи от променливи. (Член в група от променливи, които участват заедно в операция И или операция ИЛИ). Като всички останали, законите на Де Морган съществуват в две форми:

(1-5) $\overline{X_1 \vee X_2} = \overline{X_1}.\overline{X_2}$ и $\overline{X_1.X_2} = \overline{X_1} \vee \overline{X_2}$

и могат да се обобщят за повече променливи, например:

$\overline{X_1 \vee X_2 \vee X_3} = \overline{X_1}.\overline{X_2}.\overline{X_3}$

Тези теореми не следват от някакви предишни и затова трябва да бъдат доказани самостоятелно, например чрез таблици на истинност.

ЗАДАЧА: Докажете чрез таблица на истинност верността на двете форми на законите на Де Морган-формули (1-5).

2.1. ПРЕСМЯТАНЕ НА БУЛЕВИ ФУНКЦИИ

Ако логическите стойности на всички променливи са известни, тогава състоянието на функцията може да бъде определено чрез заместване на тези стойности в уравненията. Дори когато някои променливи не са известни, е възможно функцията да се опрости, а понякога и да се определи стойността й.

ПРИМЕР:

$F = \overline{A}B\left(\overline{A \vee C}\right)D =$ където A = 0, B =1, C = 1

$= \overline{0}.1\left(\overline{0 \vee 1}\right)D =$

$= 1.1.0.D = 0$

2.2. АЛГЕБРИЧНИ ТЕХНИКИ ЗА ОПРОСТЯВАНЕ

Опростяване може да бъде постигнато чрез подходящо използване на различни зависимости, закони, теореми. Необходими са някои знания и умения, за да се преобразуват успешно булеви изрази и да се постигне минимално решение. Някои примери на полезни техники са приложени по долу.

а) Групиране на членове

Членове съдържащи едни и същи променливи могат да бъдат групирани заедно използвайки дистрибутивния закон и някои вече известни зависимости.

ПРИМЕР: $A \vee AB \vee BC \vee \overline{B}C = A(1 \vee B) \vee C(B \vee \overline{B}) = A.1 \vee C.1 = A \vee C$

б) Дублиране на членове

Членове могат да бъдат повторени в един израз без да го променят. Това позволява всеки член да бъде използван в две или повече групи от членове.

ПРИМЕР:

$$AB\overline{C} \vee ABC \vee A\overline{B}C = AB\overline{C} \vee ABC \vee ABC \vee A\overline{B}C = AB(\overline{C} + C) \vee AC(B + \overline{B}) = AB \vee AC = A(B \vee C)$$

в) Разлагане на членове

Членове могат да бъдат първоначално разлагани чрез използване на зависимости, които включват нови променливи, но не променят израза. Следващото групиране може често да доведе до окончателно опростяване.

ПРИМЕР: $BCD \vee \overline{A}D \vee ABC = BCD(A \vee \overline{A}) \vee \overline{A}D \vee ABC = ABCD \vee \overline{A}BCD \vee \overline{A}D \vee ABC =$

$= ABC(D \vee 1) \vee \overline{A}D(BC \vee 1) = ABC \vee \overline{A}D$

г) Използване на законите на Де Морган

Функции могат често да бъдат опростени чрез повторно използване на законите на Де Морган върху един член или група от членове. (Използвайте закона в една от двете му форми, но не и двете едновременно)

ПРИМЕР: $\overline{\overline{ABC} \vee \overline{A}CD \vee \overline{BC}} = \overline{\overline{A} \vee \overline{B} \vee \overline{C} \vee \overline{A}CD \vee \overline{BC}} =$

$\overline{\overline{A}(1 \vee CD) \vee \overline{B} \vee \overline{C}(1 \vee B)} = \overline{\overline{A} \vee \overline{B} \vee \overline{C}} = ABC$

2.3 ОПРОСТЯВАНИЯ ИЗПОЛЗВАЩИ БУЛЕВИ ТЕОРЕМИ

За опростяване на логически изрази се използуват няколко теореми. Главните от тях са дефинирани и доказани по долу.

а) **Минимизираща теорема Minimization Theorem (Теорема за слепване);**

Ако два члена свързани с функция И или ИЛИ съдържат едни и същи променливи но се отличават по това, че една от променливите е единия член е в права, а в другия - в инверсна форма, то резултатът е само един член съдържащ общите променливи. При две променливи в член теоремата има вида:

(1-6) $(X_1 X_2) \vee (X_1 \vee \overline{X_2}) = X_1$ (дизюнктивна форма)

$(X_1 \vee X_2)(X_1 \vee \overline{X_2}) = X_1$ (конюнктивна форма)

Доказателство: $(X_1.X_2) \vee (X_1.\overline{X_2}) = X_1.X_2 \vee X_1.\overline{X_2} = X_1(X_2 \vee \overline{X_2}) = X_1.1 = X_1$

$(X_1 \vee X_2)(X_1 \vee \overline{X_2}) = X_1.X_1 \vee X_1.\overline{X_2} \vee X_2.X_1 \vee X_2.\overline{X_2} = X_1 \vee X_1(X_2 \vee \overline{X_2}) \vee 0 = X_1 \vee X_1 = X_1$

б) Поглъщаща теорема Absorption Theorem (Теорема за поглъщане);

(1-7) $X_1 \vee X_1.X_2 = X_1$ **(дизюнктивна форма)**

$X_1.(X_1 \vee X_2) = X_1$ **(конюнктивна форма)**

ЗАДАЧА: Доказателството на поглъщащата теорема в двете й форми направете сами.

в) Изобилстваща теорема Redundancy Theorem (Теорема за съкращаване)

(1-8) $X_1 \vee \overline{X_1} X_2 = X_1 \vee X_2$ **(дизюнктивна форма)**

$X_1(\overline{X_1} \vee X_2) = X_1.X_2$ **(конюнктивна форма)**

Доказателство:

$X_1 \vee \overline{X_1}.X_2 = X_1.1 \vee \overline{X_1}.X_2 = X_1(X_2 \vee \overline{X_2}) \vee \overline{X_1}.X_2 = X_1.X_2 \vee X_1.\overline{X_2} \vee \overline{X_1}.X_2 =$

$= X_1(X_2 \vee \overline{X_2}) \vee X_2(X_1 \vee \overline{X_1}) = X_1 \vee X_2$

ЗАДАЧА: Доказателството на теоремата за съкращаване в конюнктивната й форма направете сами.

г) Съгласуваща теорема Consensus Theorem (Теорема за обобщеното слепване)

(1-9) $X_1.X_2 \vee \overline{X_1}.X_3 \vee X_2.X_3 = X_1.X_2 \vee \overline{X_1}.X_3$ (дизюнктивна форма)

$(X_1 \vee X_2)(\overline{X_1} \vee X_3)(X_2 \vee X_3) = (X_1 \vee X_2)(\overline{X_1} \vee X_3)$ (конюнктивна форма)

Доказателство:

$X_1.X_2 \vee \overline{X_1}.X_3 \vee X_2.X_3 = X_1.X_2 \vee \overline{X_1}.X_3 \vee X_2.X_3.1 = X_1.X_2 \vee \overline{X_1}.X_3 \vee X_2.X_3(X_1 \vee \overline{X_1}) =$

$= X_1.X_2 \vee \overline{X_1}.X_3 \vee X_2.X_3.X_1 \vee X_2.X_3.\overline{X_1} = X_2.X_3.(1 \vee X_3) \vee \overline{X_1}.X_3.(1 \vee X_2) = X_1.X_2 \vee \overline{X_1}.X_3$

ЗАДАЧА: Доказателството на съгласуващата теорема в конюнктивната й форма направете сами.

- Изрази като сума от произведения

В таблица 4 таблично е зададена произволна функцията

таблица 4.

i	X_2	X_1	Xo	F
0	0	0	0	1
1	0	0	1	0
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	0
7	1	1	1	0

$$F = (\overline{X_2}\,\overline{X_1}\,\overline{X_o})\ V\ (\overline{X_2}\,X_1\,\overline{X_o})\ V\ (X_2\,\overline{X_1}\,\overline{X_o})$$

Всеки ред в таблицата на истинност, за които функцията има стойност лог. 1 дава произведение (И) Всички тези произведения се сумират заедно и дефинират функцията чрез единичната й стойност.

Изрази като произведения от суми

Изрази Произведения от суми може да бъдат извлечени от инверсията на израза сума от произведения. Следователно, ако НЕ F се запише във форма сума от произведения от таблицата на истинност, тогава F може да се определи във форма произведение от суми.

Таблица 4.

i	X_2	X_1	Xo	F
0	0	0	0	1
1	0	0	1	0
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	0
7	1	1	1	0

$$1 \rightarrow (\overline{X_2}.\overline{X_1}.X_o) \qquad 3 \rightarrow (\overline{X_2}.X_1.X_o)$$

$$\overline{F}=(\overline{X_2}.\overline{X_1}.X_o)V(\overline{X_2}.X_1.X_o)V(X_2.\overline{X_1}.X_o)V(X_2.X_1.\overline{X_o})V(X_2.X_1.X_o) \longrightarrow$$

$$\overline{\overline{F}}=\overline{(\overline{X_2}.\overline{X_1}.X_o)V(\overline{X_2}.X_1.X_o)V(X_2.\overline{X_1}.X_o)V(X_2.X_1.\overline{X_o})V(X_2.X_1.X_o)} =$$

$$= (X_2VX_1V\overline{X_o})\ (X_2V\,\overline{X_1}V\,\overline{X_o})\ (\overline{X_2}VX_1V\,\overline{X_o})\ (\overline{X_2}V\overline{X_1}VX_o)\ (\overline{X_2}V\overline{X_1}V\,\overline{X_o})$$

Резултатен израз произведение от суми може директно да бъде извлечен от таблицата на истинност. Всеки ред в таблицата, за който функцията има значение лог.О дава сума ((ИЛИ) ,) – Максчлен, където променливата участва в права форма, ако има 0 значение в набора и в инверсна форма ако променливата е със значение 1 в съответния набор.

ПРИМЕР: Да се определи функцията чрез нулевата и стойност от таблица 4

i	X_2	X_1	X_0	F	
0	0	0	0	1	
1	0	0	1	0	→ $(X_2 \vee X_1 \vee \overline{X_0})$
2	0	1	0	1	
3	0	1	1	0	→ $(X_2 \vee \overline{X_1} \vee \overline{X_0})$
4	1	0	0	1	
5	1	0	1	0	→ $(\overline{X_2} \vee X_1 \vee \overline{X_0})$
6	1	1	0	0	→ $(\overline{X_2} \vee \overline{X_1} \vee X_0)$
7	1	1	1	0	→ $(\overline{X_2} \vee \overline{X_1} \vee \overline{X_0})$

таблица 4

Всички тези суми (Максчленове) се умножават логически и дефинират функцията чрез нулевата й стойност като произведение от суми.

$$F = \left(X_2 \vee X_1 \vee \overline{X}_0\right)\left(X_2 \vee \overline{X}_1 \vee \overline{X}_0\right)\left(\overline{X}_2 \vee X_1 \vee \overline{X}_0\right)\left(\overline{X}_2 \vee \overline{X}_1 \vee X_0\right)\left(\overline{X}_2 \vee \overline{X}_1 \vee \overline{X}_0\right)$$

.

ПРИЛОЖЕНИЕ

Нека сега да решим един проблем, зададен по-долу чрез написаното описание.

Логически устройства трябва да управляват два светофара на натоварен преход. Последователността на изменение на светлините е показана в таблицата по долу. Тази последователност във времето се задава от управляваща (Таймерна) верига. Не се използват напреженови или индуктивни детектори за преминаващи превозни средства. Измененията в светлините са определени само от Таймерната верига.

Таймерна верига X_1	X_2	Светлини Светофар 1	Светлини Светофар 2
0	0	ЧЕРВЕНА	ЗЕЛЕНА
0	1	ЧЕРВЕНА и ЖЪЛТА	ЖЪЛТА
1	0	ЗЕЛЕНА	ЧЕРВЕНА
1	1	ЖЪЛТА	ЧЕРВЕНА и ЖЪЛТА

Първата стъпка е да начертаем таблицата, която дава състоянието на всяка от трите светлини на всеки светофар. Логическата единица отговаря на включена светлина, а логическата нула - на изключена.

Таймерна верига X_1	X_2	Светофар 1 червена	жъла	*зелена*	Светофар 2 . червена	жълт	зелена
0	0	1	0	0	0	0	1
0	1	1	1	0	0	1	0
1	0	0	0	1	1	0	0
1	1	0	1	0	1	1	0

След това цикълът се повтаря

Сега имаме множество от входове - комбинациите от таймерната верига и множество от 6 изхода. Правилото за сумата от произведения може да се приложат сега към всеки от тези изходи.

Така получаваме:

Червена 1 $= (\overline{X_1}.\overline{X_0}) \vee (\overline{X_1}.X_0)$ Червена 2 $= (X_1.\overline{X_0}) \vee (X_1.X_0)$

Жълта 1 $= (\overline{X_1}.X_0) \vee (X_1.X_0)$ Жълта 2 $= (\overline{X_1}.X_0) \vee (X_1.X_0)$

Зелена 1 $= X_1.\overline{X_0}$ Зелена 2 $= \overline{X_1}.\overline{X_0}$

Ясно е, че изразите за зелените светлини са в тяхната най-проста форма, но другите функции могат да бъдат минимизирани.

Червена 1 $= \overline{X_1}$ Червена 2 $= X_1$

Жълта 1 $= X_0$ Жълта 2 $= X_0$

Зелена 1 $= X_1.\overline{X_0}$ Зелена 2 $= X_1.\overline{X_0}$

Така дадената последователност от комбинации на променливите X_1 и Xo за светофарите се оказа сравнително проста.

Опитайте сами следващата ЗАДАЧА: От словесното описание на функцията да съставите таблица на истинност и да минимизирате функцията по познатите ви начини.

Дадени са данните:

!) Асен не говори;

2) Ваня говори само, когато Асен присъствува;

3) Соня говори винаги при всички условия дори, когато е сама;

4) Дора говори само, когато Асен присъствува.

Нека А, В, С и Д представляват присъствието на всеки човек в стаята. Определете функцията F, която представлява състоянието на тишина в стаята?

Отг. $F = \overline{A}.\overline{C} \vee \overline{B}.\overline{C}.\overline{D}$

Задачи ЛОГИКА

1. Намерете значението на логическите изрази:

1) $(1 \vee 1) \vee (1 \vee 0)$
2) $((1 \wedge 0) \vee (1 \wedge 0)) \vee 1$

2.. Оценете логическия израз като истина или лъжа
a/ $\left(\bar{a} \vee b\right) \wedge \left(a \wedge b\right)$, ako $a=1$, $b=0$
b/ $\left(a \wedge \bar{a}\right) \vee \left(\overline{a \vee b}\right)$, ako $a=1$, $b=1$
c/ $\left(a \wedge b\right) \wedge \left(\bar{a} \wedge \bar{b}\right)$, ako $a=1$, $b=0$
d/ $\left(a \wedge \bar{b}\right) \wedge \left(\overline{a \vee b}\right)$, ako $a=0$, $b=1$
e/ $\left(a \vee b\right) \wedge \left(a \wedge \bar{b}\right)$, ako $a=1$, $b=0$
f/ $\left(a \wedge \bar{b}\right) \wedge \left(\overline{a \vee b}\right)$, ako $a=0$, $b=1$
k/ $\left(\bar{a} \vee \bar{b}\right) \vee \left(a \wedge b\right)$, ako $a=1$, $b=0$
l/ $\left(a \wedge \bar{a}\right) \vee \left(\overline{a \wedge b}\right)$, ako $a=1$, $b=0$

3. Намерете кой от отговорите е равносилен на логическия израз

$A \wedge \neg(\neg B \vee C)$.

1) $\neg A \vee \neg B \vee \neg C$
2) $A \wedge \neg B \wedge \neg C$
3) $A \wedge B \wedge \neg C$
4) $A \wedge \neg B \wedge C$

4. Намерете кой от отговорите е равносилен на логическия израз

$\neg(\neg A \vee B) \vee \neg C$?

1) $(A \wedge \neg B) \vee \neg C$
2) $\neg A \vee B \vee \neg C$
3) $A \vee \neg B \vee \neg C$
4) $(\neg A \wedge B) \vee \neg C$

5. Опростете логическите изрази:

1) $(A \wedge B) \vee (A \wedge \neg B)$

2) $(A \vee B) \wedge (\neg A \vee B)$

3) $\neg(\neg X \wedge \neg Y)$

4) $(\neg X \vee Y) \wedge X$

5) $\neg(A \vee \neg B)$

6. **Опростете логическите изрази и постройте таблица за истинност. При опростяването указвайте съгласно какви закони правите преобразованията.**

2) $(A \wedge B) \vee (A \wedge \neg B)$

3) $(A \vee B) \wedge (\neg A \vee B)$

4) $\neg(\neg X \wedge \neg Y)$

5) $(\neg X \vee Y) \wedge X$

6) $\neg(A \vee \neg B)$

7. Опростете логическите функции и постройте логическите им схеми

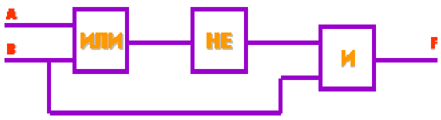
1) $F=(\neg A \vee B)(A \vee C)(B \vee C)$
2) $F=A \wedge \neg C \vee C \wedge (B \vee \neg C) \vee (A \vee \neg B) \wedge C$
3) $F=(\neg A \vee C) \wedge \neg (A \wedge C) \wedge (B \vee \neg C) \wedge \neg (B \wedge C)$

8. Даден е фрагмент от таблица с функция F:

X	Y	Z	F
1	0	0	1
0	0	0	1
1	1	1	0

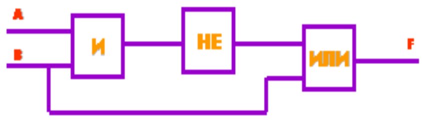
Какъв израз съответства на F?

1) $\neg X \wedge \neg Y \wedge \neg Z$ 2) $X \wedge Y \wedge Z$ 3) $X \vee Y \vee Z$ 4) $\neg X \vee \neg Y \vee \neg Z$



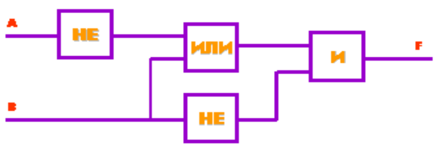
Задание 1

A	B	F
0	0	?
0	1	?
1	0	?
1	1	?



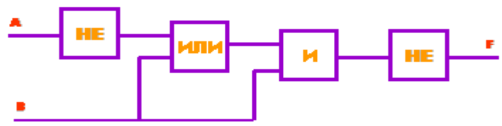
Задание 2

A	B	F
0	0	?
0	1	?
1	0	?
1	1	?



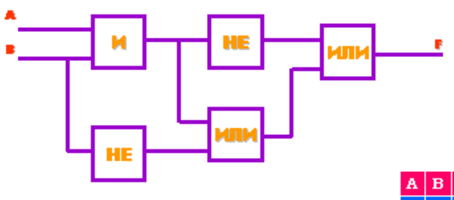
Задание 3

A	B	F
0	0	?
0	1	?
1	0	?
1	1	?



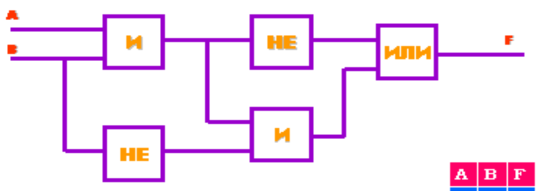
Задание 4

A	B	F
0	0	?
0	1	?
1	0	?
1	1	?



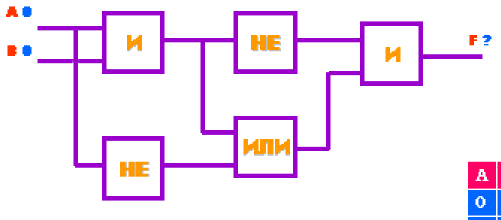
Задание 5

A	B	F
0	0	?
0	1	?
1	0	?
1	1	?



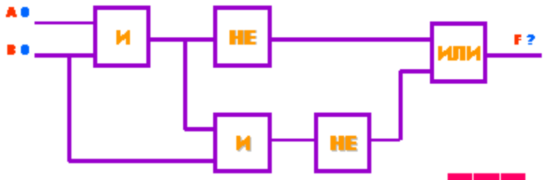
Задание 6

A	B	F
0	0	?
0	1	?
1	0	?
1	1	?



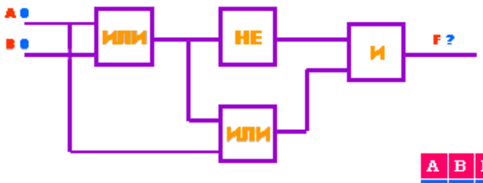
Задание 7

A	B	F
0	0	?
0	1	?
1	0	?
1	1	?



Задание 8

A	B	F
0	0	?
0	1	?
1	0	?
1	1	?



Задание 9

A	B	F
0	0	?
0	1	?
1	0	?
1	1	?

Упростите логическое выражение:

Задание 10

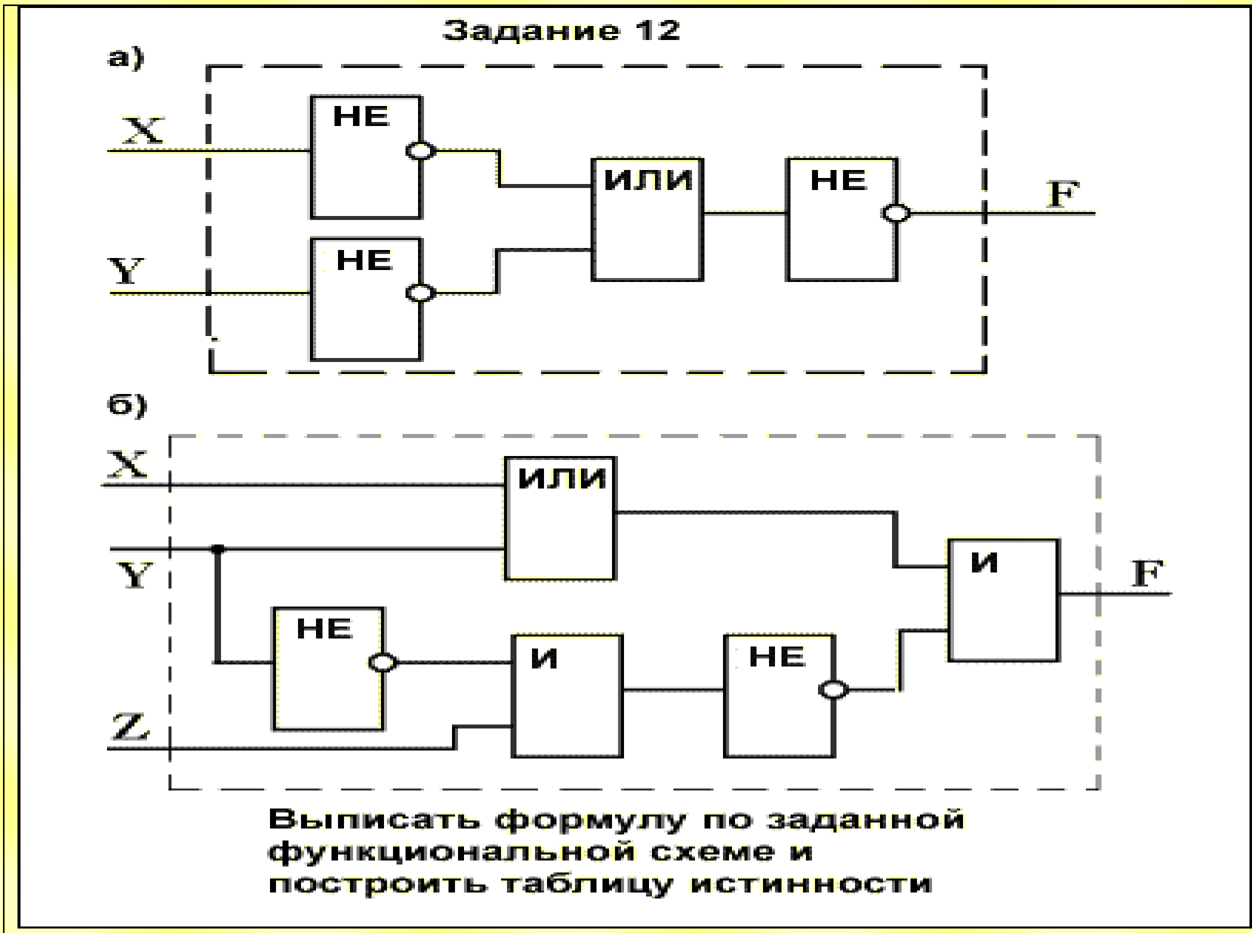
1 вариант

$$((A \,\&\, \overline{B}) \vee (A \,\&\, \overline{\overline{B}})) \vee B =$$

Задание 11

2 вариант

$$\overline{\overline{A}} \,\&\, ((\overline{\overline{B \,\&\, A}}) \vee (\overline{A} \,\&\, B)) =$$



Упражнения

zadaci

Домашнее задание

§ 3.7.1

Построить логические схемы по формулам:

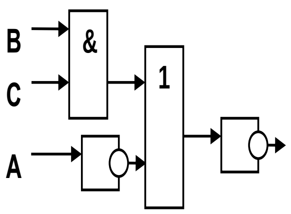
$F = \overline{A} \vee B \& C$, если A=1, B=0, C=1;

$F = (A \vee B) \& (C \vee B)$, если A=0, B=1, C=0;

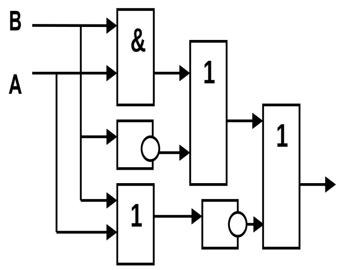
$F = (\overline{A \& B \& C})$, если A=0, B=0, C=1.

Составить логические выражения по схемам:

а)



б)



I. Алгоритъм - понятие, свойства, представяне.

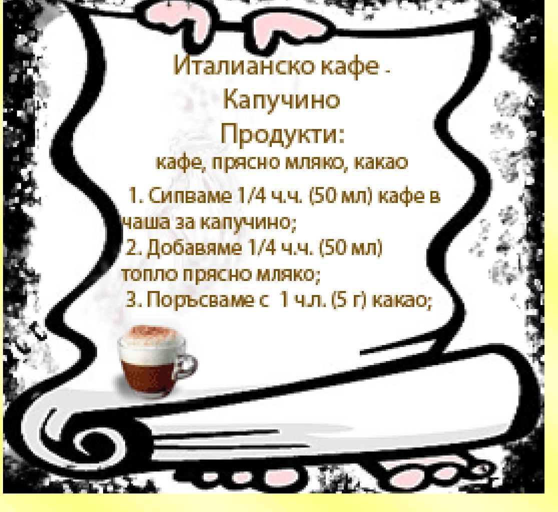
1. Понятие

Същност на алгоритмите

Как да обясним някое сложно действие на изпълнител, който умее да върши само прости неща? Трябва да представим сложното действие чрез последователност от прости действия, които изпълнителят може да извършва и които ще доведат до желания резултат.

Елементарно действие се нарича действието, което изпълнителят може да извърши самостоятелно, без допълнителни пояснения. Извършването на едно елементарно действие от изпълнителя се нарича ***стъпка***.

Алгоритъмът представлява система от указания, която задава елементарни действия и реда за изпълнението им, за да се получи определен резултат. Изпълнението на тези действия осигурява решаването на коя да е задача от определено множество.



Думата алгоритъм произлиза от името на единот най-големите учени на Средна Азия

Мухамада ибн Мус ал-Хорезми. През 1983 година се отбеляза 1200-годишнина от неговото рождение. Той е написал редица кники по аритметика и алгебра и често е наричен бащата на алгебрата, смята се че той поставя нейните основи. Ал-Хорезми показва на света с книгите си арабските цифри. През XII-XVII век в Европа е имало остра борба междуподържниците на римските цифри и тези предпочитащи арабските. Но в крайна сметка през XVII окончателните победителистават АЛГОРИТМИЦИТЕ.

2. Свойства на алгоритмите

1. **Дискретност** – всеки алгоритъм трябва да се състои от отделни, разграничени по време една от друга стъпки, всяка от които се извършва за крайно време.
2. **Детерминираност** – Резултатът от действията върху данните и посочването на следващото стъпка трябва да бъдат еднозначно определени от действията, извършвани в текущия момент.
3. **Масовост** – процедурата трябва да дава резултат на за краен брой съчетания на входните данни, а за едно потенциално безкрайно множество от входни данни.
4. **Резултатност** – процедурата трябва винаги да дава резултат, ако входните данни принадлежат на допустимото подмножество.
5. **Крайност** – процедурата трябва да завършва с резултат за краен брой стъпки.

Пример: Алгоритъм на Евклид за намиране на най-големият общ делител на две цели положителни числа p и q.

Описание:

Входни данни: p и q, p>0 и q>0;

Резултат: да се намери най-големият общ делител на p и q;

Процедура:

стъпка1: r = остатъка от p/q;

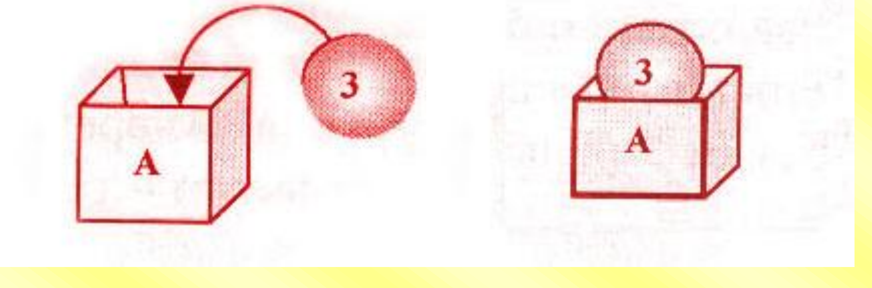
стъпка2: Ако r = 0, се предполага, че делителя е равен на q и край. В противен случай p = q, q = r и се преминава към стъпка1.

3. Видове алгоритми

Основни елементи на алгоритмите- Преди да разгледаме основните видове алгоритми нека се запознаем с някои от основните понятия при работа с алгоритми.

а)Величини - характеризират се с *име* и *тип.*Типът определя множеството от допустими стойности на величината. Стойността на величина в даден момент се нарича *текуща стойност.*

Величините се делят на *константи* и *променливи:*
- *Променлива* – Математическата представа за *променлива* е различна от понятието *променлива* при алгоритмите. В компютърните алгоритми, променливата означава елемент от паметта, чието съдържание може да се променя. На фиг. 1 имената на променливите са представени като кутии, те са отделени от своите стойности които са изобразени като топки. Резултата е показан на фиг.2 - променливата с име А добива стойност 3.



фиг.1 **фиг.2**

- *Константи* – тяхната стойност не се променя.

b)Изрази – за всеки тип величини са дефинирани различни операции, които се използват в изрази за изчисление.
В изразите величините участват със своите имена, а операциите се извършват върху стойностите им.

Основни команди за описване на алгоритми:

1. За присвояване на стойност;
2. За въвеждане и извеждане;
3. За избор на вариант (Условни команди);
4. За организиране на цикъл;
5. За обръщение към подалгоритъм.

Според последователността на действията, които се изпълняват алгоритмите биват:

1. Линейните алгоритми

Последователните (линейни) алгоритми са съставени от команди, които се изпълняват една след друга, последователно по реда на записването им. Броят на указанията е равен на броя на действията, които се извършват по време на изпълнение на алгоритъма.
Обикновено тези алгоритми реализират:

- въвеждане на входни данни (най-често от клавиатурата, но може и от файл);
- пресмятане на определени изрази;
- присвояване на резултатите на определени променливи;
- извеждане на получените резултати на екрана или на печатащо устройство

Пример: Да се определи обиколката на триъгълник със страни a, b и c.
Входни данни: a, b и c, a>0, b>0, c>0;
Резултат: P – обиколка;
Процедура:
стъпка1: задава се стойност за a;
стъпка2: задава се стойност за b;
стъпка3: задава се стойност за c;
стъпка4: P = a+b+c;

2. Разклонени алгоритми – Много често се налага да избираме между няколко възможни варианта - как да свършим определена работа, къде да идем на почивка, как да отговорим на деликатен въпрос и т. н.

В подобни случаи е необходимо да се избере един от няколко възможни варианта за действие. Този процес на избор се нарича ***вземане на решение***.

Какво решение ще вземем обикновено е свързано с това, дали е изпълнено или не някакво условие. Подобни условия се наричат ***логически условия***.

За представяне на такива условия в езиците за програмиране е въведен специален тип данни, наричан ***логически тип данни***. Множеството от стойности на логическия тип данни съдържа само две константи – истина и лъжа (на англ. ***True*** и ***False***)

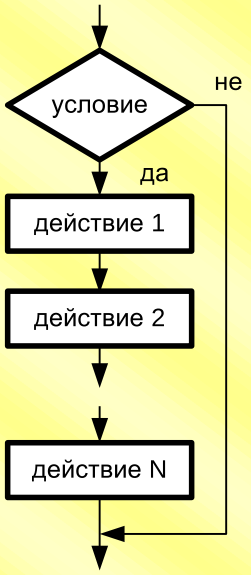
Алгоритми, които включват елементарни действия, съдържащи избор между няколко възможни варианта (т. е. моделиращи процес на вземане на решение) се наричат ***разклонени алгоритми.***

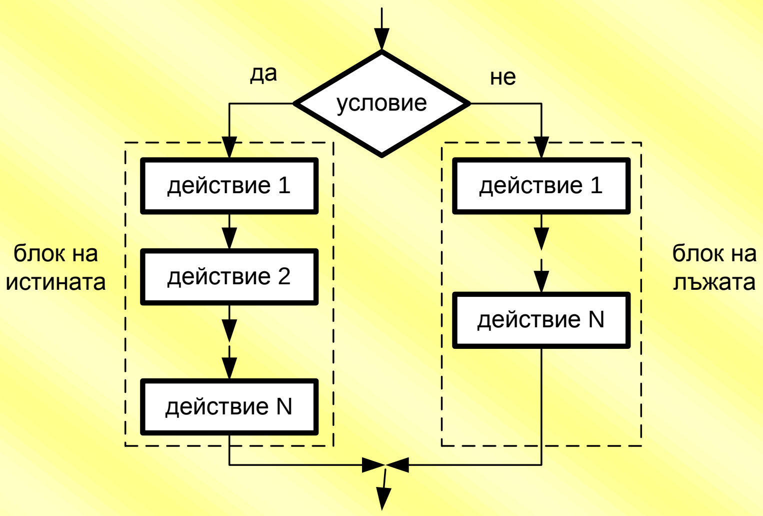
<u>Пример</u>: Деление на две числа x и y.
Величини: x, y;
Резултат: z;
Действия:
стъпка1: задава се стойност за x;
стъпка2: задава се стойност за y;
стъпка3: Ако y = 0 делението е невъзможно. В противен случай z = x/y.

Видове разклонени алгоритми

а) непълен (кратка форма): ***Ако- То***
Ако логическото условие е истина, ***То*** изпълни следните действия. В противен случай (логическото условие е лъжа) не се изпълняват никакви действия, а се преминава към действието, което е след разклонението.
Непълен алгоритъм е демонстриран в Пример 1.

б) пълен: ***Ако-То- В Противен случай***
Ако логическото условие е истина, **То** изпълни следните действия. **В противен случай** изпълни други действия.
Пълен алгоритъм е демонстриран в Пример 2.





а) Блок-схема на непълен алгоритъм   б) Блок-схема на пълен алгоритъм

3. Циклични алгоритми– Един алгоритъм е цикличен, когато при неговото изпълнение някоя редица от елементарни действия се повтаря многократно. В много задачи се налага организиране на повтарящи се действия. Такава например е задачата за намиране на сумата на редица от числа. Сумиране на голям брой числа може да се наложи, когато искате да проверите сметката си в магазина или когато класният ръководител определя средния успех на учениците в класа.

Пример: Намиране на сумата на числата от 1 до 100.
Данни: Целите числа от 1 до 100 и br;
Резултат: S – сумата;
Действия:
стъпка1: S = 0;
стъпка2: br = 1;
стъпка3: S = S+br;
стъпка4: Ако стойността на брояча е по.малка или равна на 100, стойността на брояча се увеличава с еденица и се преминава към стъпка3. В противен случай - край;

4. Описание на алгоритмите

Съществуват три подхода при записване на алгоритмите: текстови, графичен и посредством формални езици. Всеки един от трите подхода има своите предимства и недостатъци.

a) Текстови подход

При текстовия подход се описват словесно последователност от стъпки, които трябва да се изпълнят за да се реши задачата.

Всички стъпки са номерирани според реда на тяхното изпълнение. Ако се налага нарушаване на този ред (за реализация на разклонения или цикли) се описва изрично коя е следващата стъпка.

Пример:
Намиране корените на едно квадратно уравнение $y = ax^2+bx+c$.
Стъпки:

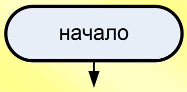
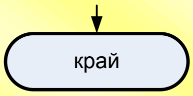
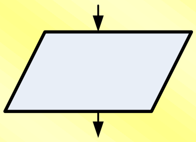
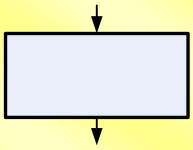
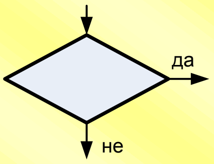
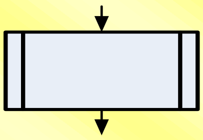
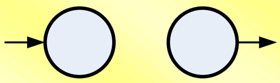
1. Изчислява се $D=b^2-4ac$.

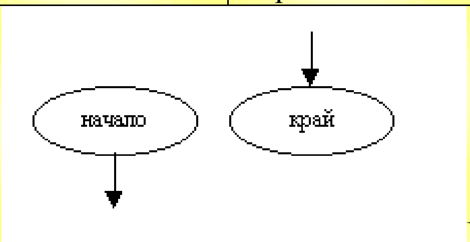
2. Проверява се знака на D. Ако D<0 се преминава към точка 3. В противен случай се изчислява x1= (-b+√D) / (2a), x1= (-b-√D) / (2a).
3. Прекратява се изчисляването.

b) Графично представяне - блокови схеми

Всяка стъпка се представя като възел в един насочен граф. Потокът на управление се оказва от свързващите стрелки.

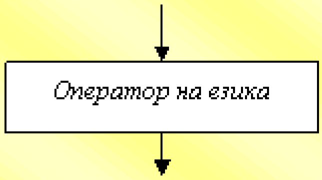
Видове възли според ANSI X3.5-1970 стандарта:

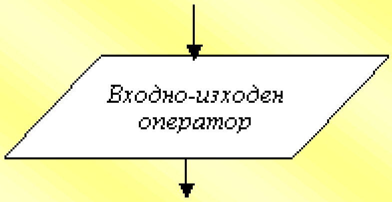
Символ	Предназначение
	**Блок за начало на блок-схемата**
	**Блок за край на блок-схемата**
	**Блок за вход или изход** В блока за вход се изреждат величините, които са необходими за изпълнение на алгоритъма. В блока за изход се изписва крайният резултат – целта на алгоритъма
	**Блок за обработка (изчисления)** Изчисляват се изрази и стойността им се присвоява на променлива. Съдържанието на блока има вида: ***Променлива : = израз*** *": =" е знак за присвояване*
	**Блок за анализ** В блока се записва логически израз – условие, чиято стойност определя кои блокове да бъдат изпълнени. В зависимост от това дали е изпълнено условието или не, изчислителният процес се разклонява на две посоки
	**Блок за подалгоритъм** В него се изписва името на алгоритъма и величините (параметрите), за които трябва да се изпълни
	**Блокове за преход (връзка)** Използват се за свързване на отдалечени блокове. Възможен е само един входящ с дадена буква или цифра, докато броят на изходящите със същата буква или цифра е неограничен
	**Свързваща стрелка** Осъществява връзката между блоковете и е прието да е хоризонтална или вертикална. Допуска се да бъде начупена

- 

Едновходови:

- Двувходови:





- Тривходови:

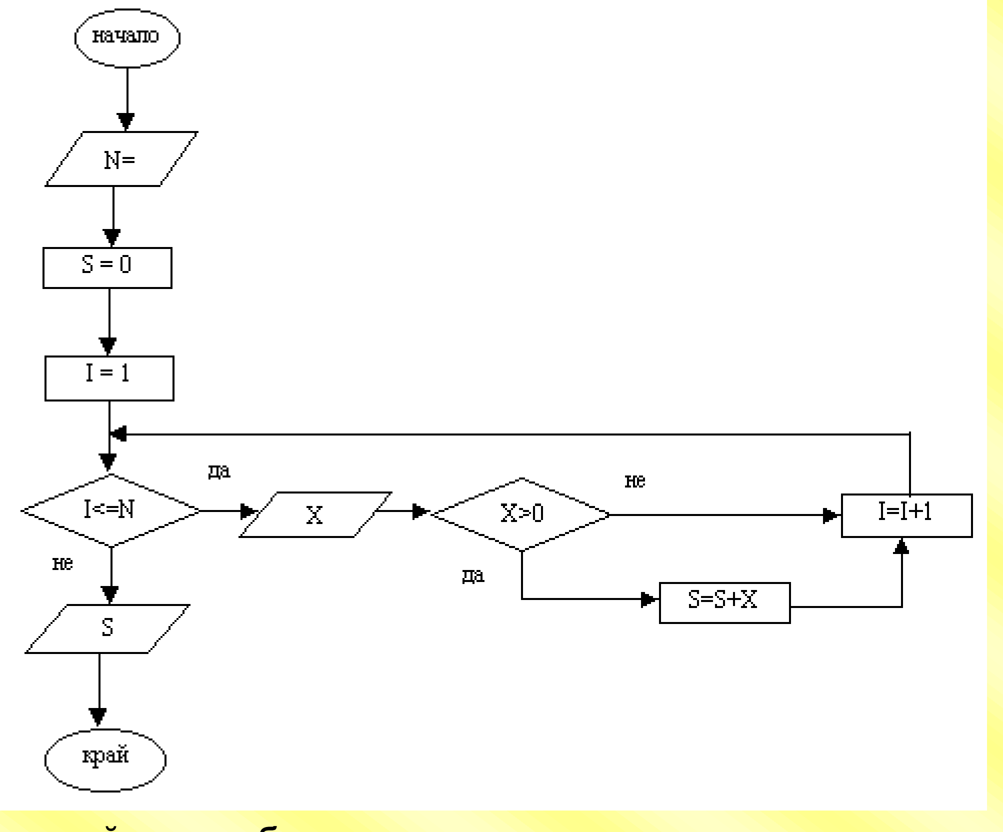
b) Чрез формални езици (езици за програмиране)

Пример

Да се напише програма, която въвежда n на брой числа от клавиатурата и намира сумата само на онези от тях, които са положителни. Описана е с блоковата схема на фиг.5.

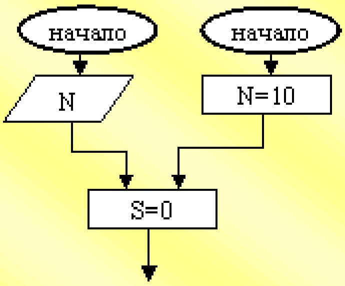
```
#include <iostream.h>

void main(void)

{

 int i,n;

 float x,s;

 do{

   cout<<"n=";  cin>>n;

 }while (n<1 || n>1000);



 for(s=0,i=1;i<=n;i++)
```

```
 {
     cout<<"x=";  cin>>x;
     if(x>0) s+=x;
 }
 cout<<"s="<<s<<"\n";
}
```



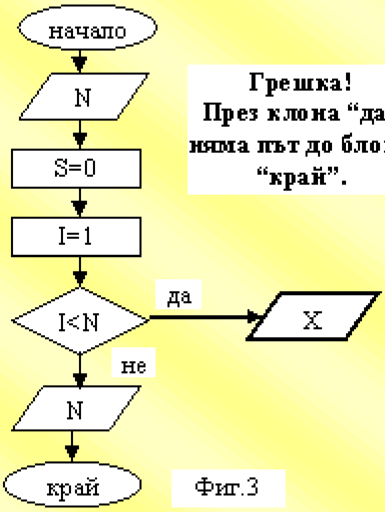
Основни свойства на блоковите схеми

1. Точно едно "начало" и поне един "край".
2. През всеки един възел да има път от началото към края.
3. Да не се позволява зацикляне.



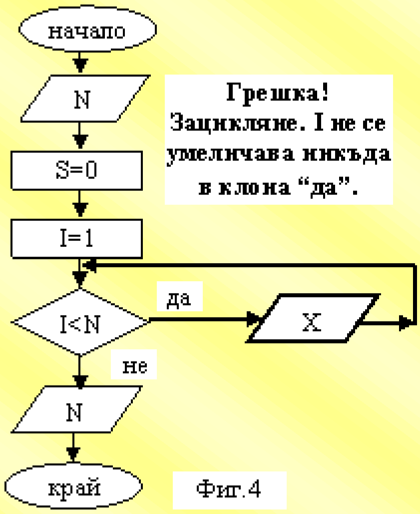
Фиг.2

Грешка!
Два входа



Грешка!
През клона "да" няма път до блока "край".

Фиг.3



Грешка!
Зацикляне. I не се умелнчава никъда в клона "да".

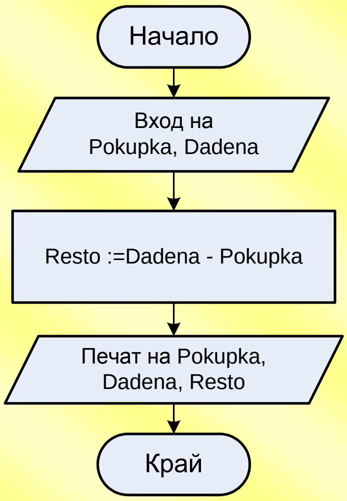
Фиг.4

Примери чрез блокови схеми:

→линейни

1 Алгоритъм, който при зададени стойност на покупката и сума, дадена от купувача, извежда информация за рестото

Означаваме стойността на покупката с променливата Pokupka, сумата, дадена от купувача – Dadena, а рестото – Resto

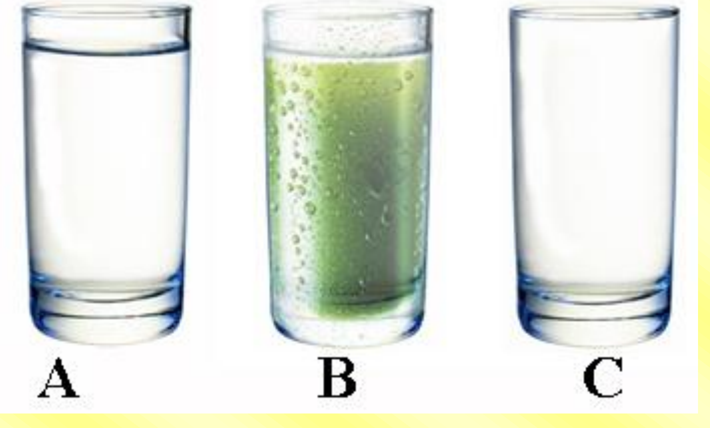


Задача за самостоятелна работа:

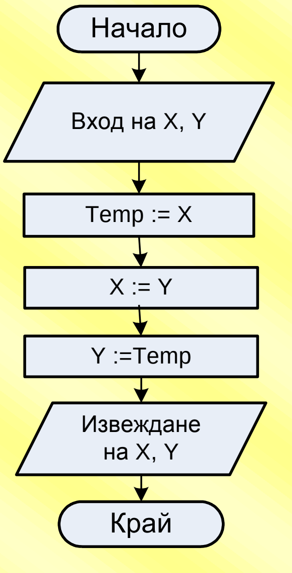
Алгоритъм, който при зададена цена на стоката изчислява цената с включено ДДС (ДДС се приема за 20%).

2 Алгоритъм, който разменя стойностите на две променливи

За да разберем най-лесно какстава размяната, нека да си
представим че променливата A е чаша пълна с вода,
а променливата B е чаша пълна със натурален сок.
За да разменим съдържанието надвете чаши, най-добре е да намерим
трета, която е празна.След това в празната чаша C, изсипваме съдържанието от чаша A - водата. Сега празна ни е чаша A, в нея изсипваме съдържанието на чаша B - натурален сок. Празната чаша сега е B. И накрая водата от чаша C изсипваме в чаша B. Така съдържанието на A и B вече са разменено.



Означаваме променливите с X и Y. Въвеждаме спомагателна променлива Temp, която съхранява междинна стойност.



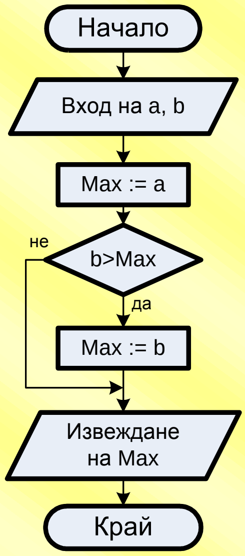
Задача за самостоятелна работа:

Алгоритъм, който разменя циклично стойностите на три променливи. Например, ако X=1, Y=2, Z=3, в резултат ще се получи X=2, Y=3, Z=1

→ разклонените алгоритми

Пример 1: Да се състави алгоритъм за намиране на по-голямото от две реални числа a и b.

Решение:



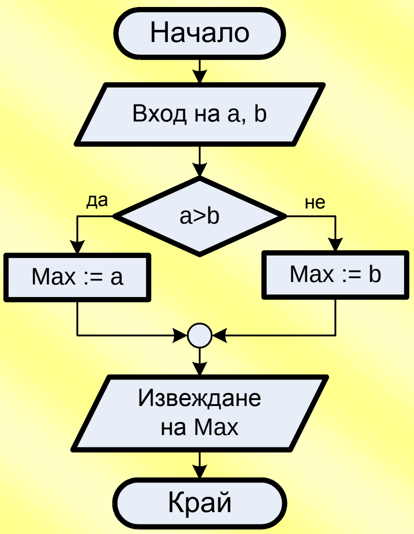
Входни данни: числата a и b

Изходни данни: по-голямото число Max.

Словесно описание на алгоритъма:

1. Присвоява се стойността на числото a на променливата Max: Max:=a
2. Проверява се дали b е по-голямо от Max. Ако b>Max, преминава се към стъпка 3. В противен случай се преминава към стъпка 4.
3. Присвоява се стойността на числото b на променливата Max: Max:=b.
4. Извежда се стойността на Max.

Пример 2: Задачата от Пример 1 може да се реши и чрез прилагане на пълен разклонен алгоритъм



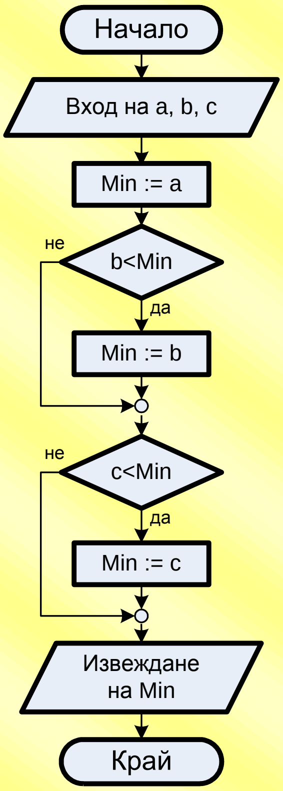
Решение:

Входни данни: числата a и b

Изходни данни: по-голямото число Max.

Словесно описание на алгоритъма:

1. Проверява се дали a е по-голямо от b. Ако a>b, преминава се към стъпка 2. В противен случай се преминава към стъпка 3.
2. Присвоява се стойността на числото a на променливата Max:
Max:=a.
3. Присвоява се стойността на числото b на променливата Max: Max:=b.
4. Извежда се стойността на Max.

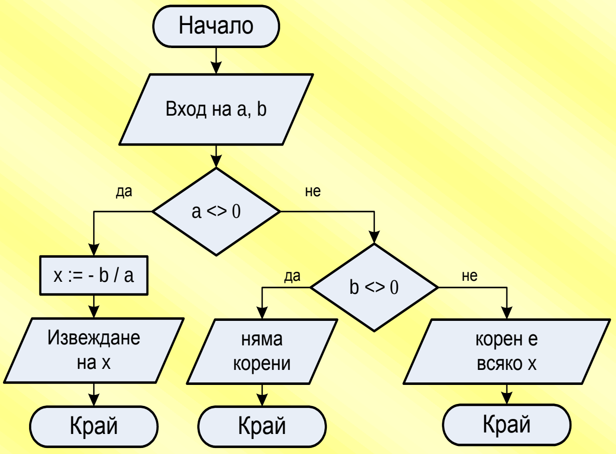


Пример 3: Да се състави алгоритъм за намиране на най-малкото от 3 дадени реални числа.

Решение:
Входни данни: числа a, b и c
Изходни данни: най-малкото от трите числа Min.
1. Присвоява се стойността на числото a на променливата Min.
2. Проверява се дали b е по- малко от Min. Ако b<Min, преминава се към стъпка 3. В противен случай се преминава към стъпка 4.
3. Присвоява се стойността на числото b на променливата Min. Преминава се към стъпка 4.
4. Проверява се дали c е по- малко от Min. Ако c<Min, преминава се към стъпка 5. В противен случай се преминава към стъпка 6.
5. Присвоява се стойността на числото c на променливата Min.
6. Извежда се стойността на Min.

Пример 4: Да се състави алгоритъм за намиране на корените на линейното уравнение: a.x + b = 0.

Решение:
Входни данни: коефициентите a и b
Изходни данни: корените на уравнението.



Словесно описание на алгоритъма:
1. Проверява се дали коефициентът a ≠ 0. Ако a ≠ 0, преминава се към стъпка 2. В противен случай се преминава към стъпка 3.
2. Уравнението има единствен корен x = - b/a
3. Проверява се дали коефициентът b ≠ 0. Ако b ≠ 0, преминава се към стъпка 4. В противен случай се преминава към стъпка 5.
4. Тъй като a = 0 и b ≠ 0, уравнението няма корени.
5. Тъй като a = 0 и b = 0, уравнението има безброй много корени в областта на реалните числа ($\forall x \in R$).

Виждат се следните особености:
1. Разклоненият алгоритъм може да има повече от един край.
2. В един алгоритъм може да съществуват повече от едно логически условия.

Пример 5: Да се състави алгоритъм за намиране на корените на квадратното уравнение:

$a.x^2 + bx + c = 0.$

Решение:

Входни данни: коефициентите a, b, c

Изходни данни: корените на уравнението.

Словесно описание на алгоритъма:

1. Проверява се дали коефициентът a = 0. Ако a = 0, преминава се към стъпка 2. В противен случай се преминава към стъпка 3.
2. Изпълнява се подалгоритъм за решаване на линейно уравнение $bx + c = 0$.
3. Изчислява се дискриминантата $D := b^2 - 4.a.c$
4. Проверява се D<0. Ако D<0, то се извежда съобщение „Няма корени в областта на реалните числа.” Край на алгоритъма.

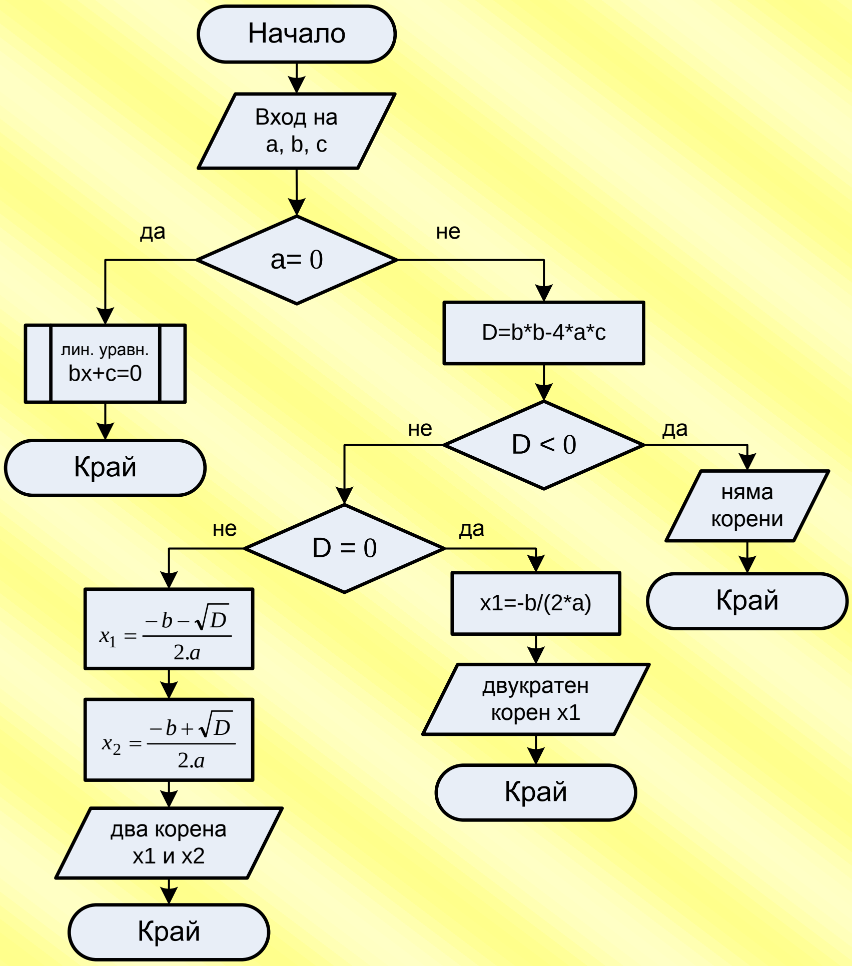
В противен случай се преминава към стъпка 5.

5. Проверява се D = 0. Ако D = 0, коренът е двукратен и стойността му е $x_{1,2} = \frac{-b}{2.a}$.

Край на алгоритъма.

В противен случай се преминава към стъпка 6.

6. Корените на уравнението са два: $x_1 = \frac{-b+\sqrt{D}}{2.a}$; $x_2 = \frac{-b-\sqrt{D}}{2.a}$. Край на алгоритъма.



Задачи за самостоятелна работа:

1. Да се състави алгоритъм за подреждане по големина във възходящ ред на 3 числа: x, y и z.

   Упътване: Сравнете последователно x и y, y и z, x и z.

2. Да се състави алгоритъм, който определя в кой квадрант лежи точката A с координати x и y. Предполага се, че точката не лежи на никоя от координатните оси, т.е $x \neq 0$ и $y \neq 0$.

   Упътване: В кой квадрант лежи точката A зависи от знаците на x и y.

3. Решете задача 2, като ако т. A лежи на някоя от координатните оси, се извежда подходящо съобщение.

4. Да се състави алгоритъм, който въвежда три реални числа a, b, c и проверява дали съществува триъгълник със страни a, b, c. Ако съществува, да се определи видът му: разностранен, равнобедрен или равностранен.

   Упътване: За да съществува един триъгълник, трябва всяка страна да е положителна, т.е. да е истина съждението: a>0 И b>0 И c>0 и всяка страна да е по-малка от сумата на останалите две страни.

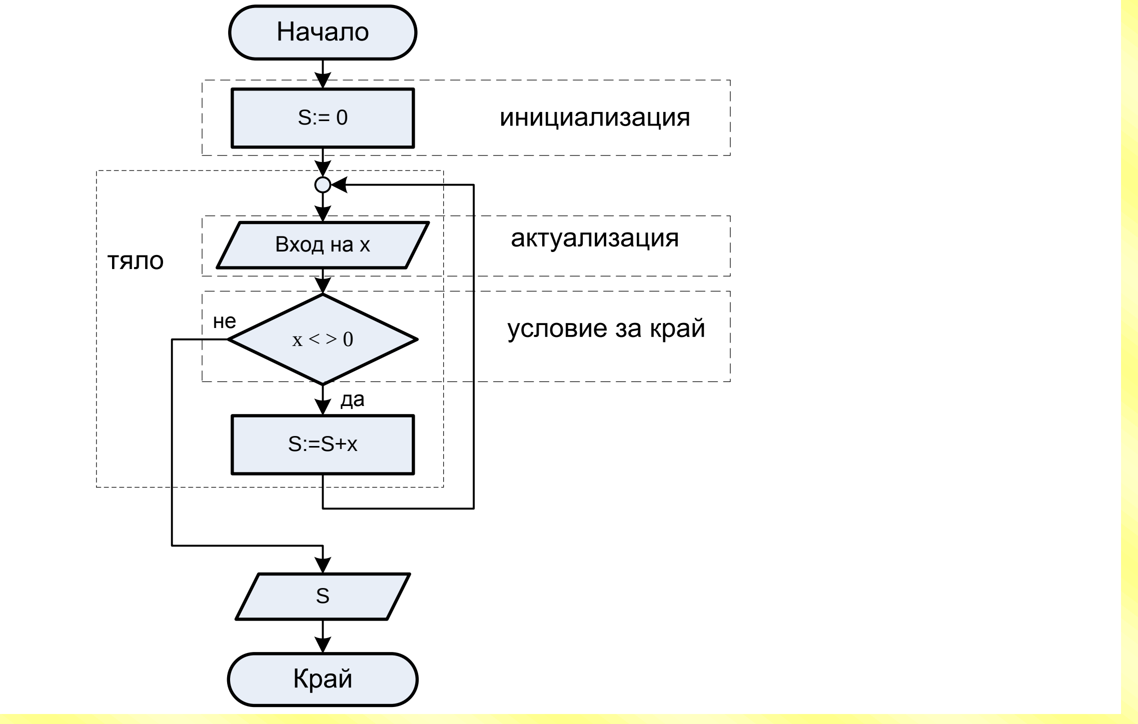
→циклични алгоритми

Пример 1: Алгоритъм за определяне на сума

Предложеният алгоритъм решава задачата за намиране на сумата на числа, въвеждани последователно, като сумирането завършва, когато се въведе числото 0. Сумата на числата се натрупва последователно (след всяко въвеждане) в S.

Следващото словесно описание представя цикличен алгоритъм за решаване на задачата:

1. S := 0.
2. Въведете поредното число x.
3 Ако $x \neq 0$, то s :=s + x, в противен случай изпълнете стъпка 5.
4. Изпълнете стъпка 2.
5. Съобщете стойността на S (като резултат).
6. Прекратете работа.



Елементи на цикъла.

а) **тяло на цикъла -** групата от повтарящи се действия в цикличните алгоритми

В примера тялото на цикъла съдържа действията, включени в стъпки 2, 3 и 4 (фиг .1).

Всяко изпълнение на тялото на цикъла се нарича ***итерация***. Задължително изискване е повторенията да са краен брой.

б) **условие за край на повторенията** – определя кога ще завърши цикълът

В горния алгоритъм условието за край е x = 0 (проверява се на стъпка 3).

в) **управление на повторенията (актуализация)** – действие в някоя от стъпките на цикъла, което евентуално ще предизвика удовлетворяване на условието за край. В противен случай цикълът може да продължи безкрайно.

В примера управлението на повторенията е определено със стъпка 2, защото нововъведената стойност на x може да е 0 и да предизвика приключване на изпълнението на алгоритъма.

г) **инициализация (подготовка)** – задава се начална стойност на някои променливи, участващи в цикъла.

В примера стъпка 1 задава правилната начална стойност на сумата s, която е предпоставка за правилен резултат в края на алгоритъма, т.е. получаване сумата на въведените числа.

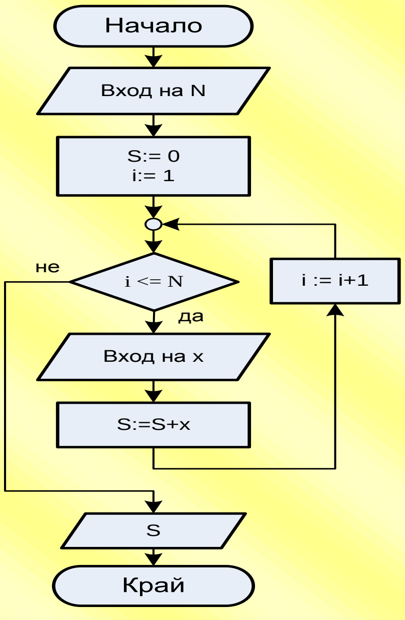
Четирите основни елемента на всеки цикъл: инициализация, тяло, управление на повторенията и условие за край, трябва внимателно да бъдат обмислени и проектирани, независимо по какъв начин ще се записва цикличният алгоритъм – словесно, чрез блок схема или чрез оператор за цикъл.

Видове циклични алгоритми

В зависимост от това дали е предварително известен броят на повторенията на цикъла цикличните алгоритми биват:

- **итеративни** - броят на повторенията на цикъла не е известен предварително; този брой зависи от обработваните данни;
  Итеративен е алгоритъмът от Пример 1.
- **индуктивни** - броят на повторенията на цикъла е предварително известен; този брой се задава чрез параметър (управляваща променлива), наречен брояч.
  Индуктивен са алгоритмите от Примери 2 и 3.

Пример 2: Да се намери сумата на N числа, въвеждани последователно от клавиатурата.



Следващото словесно описание представя цикличен алгоритъм за решаване на задачата:

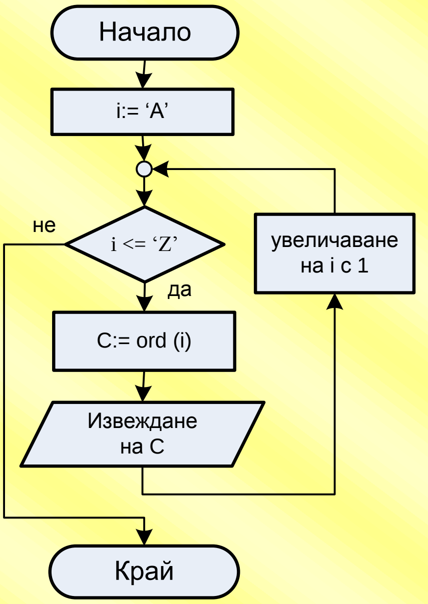
Въвежда се брояч i. Неговата начална стойност е 1, а крайната - N (колкото е броят на числата).

1. S:= 0; i:=l
2. Проверява се дали броячът i е по-малък или равен на броя на числата N. Ако това е вярно, изпълнява се стъпка 3, в противен случай се изпълнява стъпка 7.
3. Въведете поредното число x.
4. s :=s + x
5. i:= i+1
6. Изпълнете стъпка 2.
7. Съобщете стойността на S (като резултат).
8. Прекратете работа.

Особености на индуктивните цикли:

1. Управляващата променлива е от дискретен тип (цяло число или символ).
2. Проверката за край на цикъла се извършва преди изпълнението на тялото, което означава, че цикълът може да не се изпълни нито веднъж.
3. Началната и крайната стойности на управляващата променлива (брояча) се изчисляват еднократно в началото на цикъла.
4. Промяната на управляващата променлива в операторите за цикъл, използвани в езиците за програмиране, се извършва автоматично и не се препоръчва нейната стойност да се променя от програмиста.

Пример 3: Да се изведат кодовете на главните латински букви



фиг. 3 Блок схема на алгоритъм за извеждане кодовете на главните латински букви

Въвежда се брояч i от символен тип. Неговата начална стойност е 'A', а крайната – 'Z'.

1. i:= 'A'
2. Проверява се дали броячът i е по-малък или равен на 'Z'. Ако това е вярно, изпълнява се стъпка 3, в противен случай се изпълнява стъпка 7.
3. Изчислява се кодът на i чрез вградена функция.
4. Извежда се кодът на i.
5. Преминава се към следващия по ред символ.
6. Изпълнете стъпка 2.
7. Прекратете работа.

Забележка: При данни от символен тип се работи с кодовете им. При сравняване на две променливи по-малка е тази, която има по-малка стойност на кода. Символът, който е след даден символ, има код по-голям с 1 от кода на дадения символ. Например в ASCII таблицата, кодът на 'A' е 65, на 'B' е 66 а на 'C' е 67.

В зависимост от това дали условието за край на цикъла се проверява преди изпълнението на тялото на цикъла или след него, итеративните циклични алгоритми биват:

- **цикли с предусловие** - проверката на условието се извършва преди тялото на цикъла, което означава, че тялото може да не се изпълни нито веднъж;
- **цикли със следусловие-** проверката на условието се извършва след тялото на цикъла, което означава, че тялото ще се изпълни поне веднъж.

И в двата случая в тялото на цикъла трябва да има действия, които да водят до промяна на поне един от параметрите, участващи в условието за край на повторенията. В противен случай ще се получи зацикляне (безкрайно повторение). Промяната на параметрите (за разлика от алгоритмите с брояч) е грижа на програмиста.

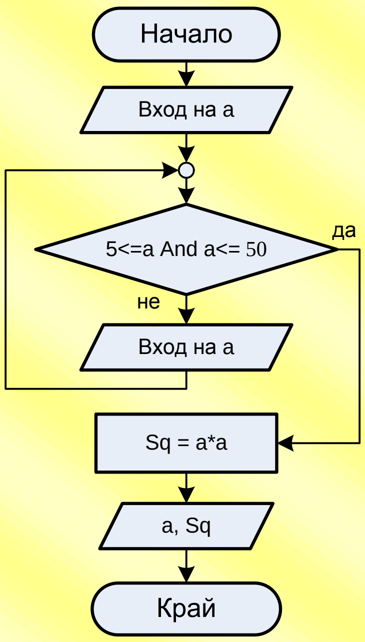
При циклите с предусловие обикновено условието за край трябва да се инициализира преди влизане в цикъла.

Итеративните циклични алгоритми често се използват за проверка на коректността на въведените данни.

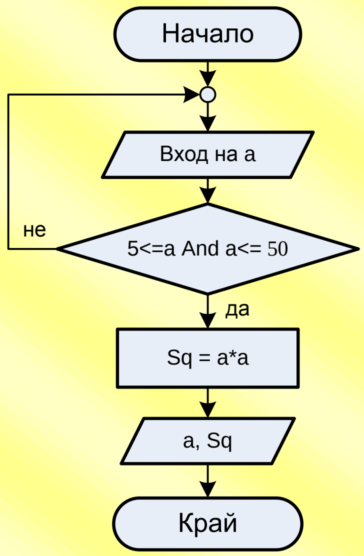
Пример 4: От клавиатурата се въвежда число n в интервала 5<= n <= 50. Ако числото не е в този интервал, да се изисква ново въвеждане. При коректно въведени данни да се изчисли и изведе квадратът на числото.

Решение:

На фиг. 4а е показана блок-схемата на алгоритъм с предусловие, а на фиг. 4б - алгоритъм със следусловие. Вижда се, че в случая по-подходящ е алгоритъмът със следусловие.

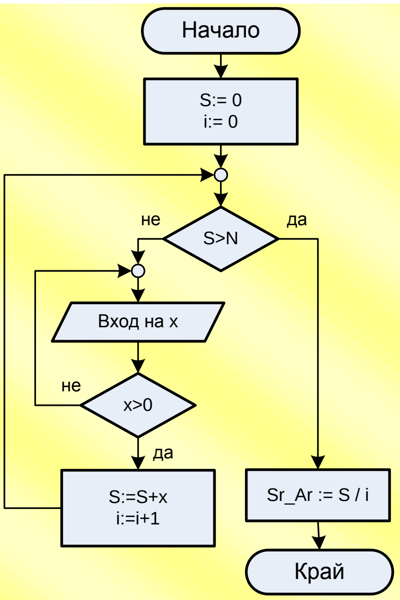


а) алгоритъм с предусловие



б) алгоритъм със следусловие

Пример 5: Да се намери средноаритметичното на положителни числа, въвеждани от клавиатурата. Прекратяването на въвеждането да стане, когато сумата на въведените числа надвиши предварително зададено число.



В показания на фиг .5 алгоритъм:
1. Прави се инициализация на сумата и броя на въведените числа: S:=0; i:= 0;
2.Прави се проверка за коректността на данните – докато не се въведе положително число не се преминава към следващата стъпка.
3. Във всяка итерация се увеличава сумата с въведеното число x и броя на числата с 1.
4. Когато S стане по-голямо от предварително зададено число N изпълнението на цикъла се прекратява и се извършва изчисляване на средноаритметичното, като се раздели сумата на броя им.

Задачи за самостоятелна работа:

1. По време на голямото веждучасие гладен ученик отишъл в стола за да си купи 5 сандвича. Пари за такова количество сандвичи обаче нямал в себе си. Ученикът, който бил до него, го посъветвал да използва такъв алгоритъм:

   a. Докато не изчезне чувството за глад да повтаря.
   b. Купува 1 сандвич.
   c. Край на цикъла.
   d. Яде сандвич.

Намерене грешките и начертайте блок схема на правилният алгоритъм, така че ученика да не си тръгне гладен от столовата.

2. Да се състави алгоритъм, който въвежда от клавиатурата естествено число n. Алгоритъмът да изчислява и извежда:

а) сумата на всички цели числа от 1 до n. Примерен вход: 5 Примерен изход: 15

б) произведението на всички цели числа от 1 до n (n!=1*2*3*. . . *n се нарича n факториел). Примерен вход: 5 Примерен изход: 120

3. Да се състави алгоритъм, който въвежда от клавиатурата естествено число n и реално число a. Алгоритъмът да изчислява и извежда a^n.

Примерен вход: 3 2.5 Примерен изход: 15.625

4. Да се състави алгоритъм, който въвежда от клавиатурата естествено число k. Алгоритъмът да извежда всички трицифрени числа, сумата от цифрите на които е равна на числото k.

Упътване*: За изреждане на всички трицифрени числа се използват три вложени цикъла: най-външния обхожда числата от 1 до 9 (цифрата на стотиците), а двата вътрешни – числата от 0 до 9 (за цифрите на десетиците и единиците). За всяка комбинация от цифри се проверява дали е равна на числото k. Трицифреното число се получава като: ЦифраСтотици*100+ЦифраДесетици*10 + ЦифраЕдиници.*

5 Да се състави алгоритъм, който въвежда от клавиатурата целите числа m и n, (m<n). Алгоритъмът да извежда всички числа в интервала [m, n], които са кратни на 5. Направете проверка за коректност на входа (m<n).

Упътване*: I начин: Изреждат се чрез цикъл всички цели числа в интервала [m, n]. За всяко от тях се проверява дали е кратно на 5 чрез функцията MOD (модул). Ако е кратно на 5, модулът на числото по 5 е равен на нула. В Pascal това се записва: число mod 5 =0. В С++ се записва така: число%5==0*

II начин: Намира се най-близкото по-голямо или равно число на m, което е кратно на 5. Следващите кратни числа се получават чрез цикъл, като на всяка итерация се добавя 5, докато се надвиши числото n.

Какви са предимствата на всеки от двата начина? Помислете за свой собствен начин.

6. Да се състави алгоритъм, който намира и извежда всички трицифрени числа, които нямат в записа си цифра нула и са кратни на всяка своя цифра.

7. Да се състави алгоритъм, който извежда на екрана всички четирицифрени числа, сумата от цифрите, на които е двуцифрено четно число.

Упътване: *Едно число е четно, ако модулът на числото по 2 е равен на нула (вж. зад 4). Едно цяло число е двуцифрено, ако е по-голямо от 9 и по-малко от 100.*

8. Да се състави алгоритъм, който намира и извежда всички четирицифрени числа, които имат поне две равни цифри.

Упътване: *Виж зад. 3*

9 Да се състави алгоритъм, който намира и извежда всички трицифрени числа, сумата от цифрите на които е просто число.

10. Да се състави алгоритъм, който въвежда от клавиатурата естествено число n и n на брой символи. Алгоритъмът да извежда броя на тези от въведените символи, които са малки латински букви. Примерен вход: 6 w ! 5 a B k Примерен изход: 3

Упътване: *За всеки символ може чрез функция да се получи кода и да се провери дали попада в определен интервал (Виж Пример 3 от урока). Малките латински букви имат кодове от 97 до 122 включително. Може да се проверява и директно: C >='a' And C <='z'*

11. Да се състави алгоритъм, който въвежда от клавиатурата цифра k, естествено число n и n на брой естествени числа. Алгоритъмът да извежда броя на онези числа от въведените, които са k-цифрени.

Примерен вход: 2 5 Примерен изход: 3

3 25 33 134 18

12. Да се състави алгоритъм, който въвежда от клавиатурата цели положителни числа. За край на въвеждането служи числото нула. Алгоритъмът да намира и извежда сумата на четните и броя на нечетните от въведените числа.

Примерен вход: 2 4 18 33 96 7 0 Примерен изход: 120 2

Примерен вход: 1 5 33 9 7 0 Примерен изход: 0 5

13. Да се състави алгоритъм, който въвежда от клавиатурата цели положителни числа. За край на въвеждането служи числото нула. Алгоритъмът да намира и извежда най-голямото от тях, което е нечетно число. В случай че въведените числа са само четни, да се изведе подходящо съобщение.

Примерен вход: 2 4 18 33 96 7 0 Примерен изход: 33

Примерен вход: 6 4 18 16 0 Примерен изход: Не сте въвели нечетни числа

14. Върху числовата ос до 100 см са нанесени деления през 1 см. Да се състави алгоритъм, който въвежда от клавиатурата естествено число n (2 <= n <= 100) и n на брой двойки цели числа, които задават краищата на отсечки върху числовата ос. Алгоритъмът да извежда дължината на най-голямата и най-малката зададена отсечка.

Упътване: *Дължината на отсечката се получава като се извадят числата, които задават краищата ѝ. Полученото число се взема с модул, за да бъде положително.*

15. Да се състави алгоритъм, който въвежда от клавиатурата цяло число n и след него n на брой цели числа. Алгоритъмът да проверява има ли измежду тях последователни равни числа и извежда съобщение Yes или No в съответния случай.

Примерен вход: 7 2 4 33 33 96 7 0 Примерен изход: Yes

Задачи АЛГОРИТМИ

1. Дадено е числото **a.** Като използвате само умножение получете:
   1.1. a^4 с две операции.
   1.2. a^6 с три операции.
   1.3. a^7 с четири операции.
   1.4. a^8 с три операции.
   1.5. a^9 с четири операции.
   1.6. a^{10} с четири операции.
   1.7. a^{13} с пет операции.
   1.8. a^{15} с пет операции.
   1.9. a^{21} с шест операции.
   1.10. a^{28} с шест операции.
   1.11. a^{64} с шест операции.

2. Дадено е двуцифрено число. Намерете:
   2.1. Цифрата на десетиците му
   2.2. Цифрата на единиците му
   2.3. Сумата от цифрите му
   2.4. Произведението на цифрите му
   2.5. Числото, което се получава, като обърнем местата на цифрите му

3. Дадено е трицифрено число. Намерете:
   3.1. Цифрата на десетиците му
   3.2. Цифрата на единиците му
   3.3. Цифрата на стотиците му
   3.4. Сумата от цифрите му
   3.5. Произведението на цифрите му
   3.6. Числото, което ще се получи при четене на числото отдясно наляво
4. Дадено е трицифрено число
   4.1. Намерете числото, което се получава като първата му цифра се премести накрая
   4.2. Намерете числото, което се получава като последната му цифра се премести в началото
   4.3. Намерете числото, което се получава като сменим местата на първата и втората му цифра
   4.4. Намерете числото, което се получава като сменим местата на втората и третата му цифра

5. Определете значението на променливата ***c*** след изпълнение на фрагмента от програма:

```
a:=5 a:=a+6 b:= –a
 c:=a–2*b
```

1) c = –11   2) c = –11   3) c = 27   4) c = 33

6. При сделка купувач купил 12 еднакви по вид златни монети. Станало му известно че една от монетите е фалшива и се различава само по теглото от истинските. За определяне на фалшивата той номерирал монетите с числа от 1 до 12 и провел 3 измервания на везна. Обозначил с V_i – всяка монета с номер i, той записал резултатите :
$V_1+V_2+V_7+V_8>V_3+V_4+V_5+V_6$;
$V_1+V_5+V_8+V_9 <V_2+V_7+V_{10}+V_{11}$;
$V_1+V_6+V_7+V_{12} <V_2+V_4+V_9+V_{10}$.

Кой е номера на фалшивата монета?

1) 1 2) 2 3) 7 4) няма фалшива

7. В един клас има 27 ученика. 13 имат куче, 11 – котки, а 9 – папагали. Едновременно куче, котка и папагал имат 2-ма човека, куче и котка – имат 6 човека, котка и папагал – 4 човека, куче и папагал – 5 човека. Колко ученика нямат животни?

преговор

Бройни системи

1. Превърнете в десетична бройна система числата: а) $1010111_{(2)}$; б) $1423_{(8)}$; в) $1A4_{(16)}$;

2. Превърнете в двоична бройна система числата: а) $125_{(10)}$; б) $1423_{(8)}$; в) $1A4_{(16)}$;

3. Превърнете в осмична бройна система числото: $1010111_{(2)}$
4. Превърнете в шестнадесетична бройна система числото: $1011010111_{(2)}$

Булева алгебра

1. Докажете чрез таблица на вярност закона: (p.q) v r = (p v r) . (q v r) – 10 т.

2. Функцията F (A, B, C, D) е зададена в следната таблица:

A	0	0	0	0	0	0	0.	0	1	1	1	1	1	1	1	1
B	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
C	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
D	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
F	0	0	1	1	0	0	0	0	1	0	1	0	1	0	0	1

а) Представете аналитично функцията F в дизюнктивна нормална форма - 10 т.;

б) Минимизирайте функцията F чрез използване на аналитичен или графичен метод – 20 т.;

в) Съставете по минимизираната функция логическа схема чрез използване на елементи НЕ, И, ИЛИ – 10 т.;

Алгоритми

1. Начертайте блок схема на алгоритъм, който разменя циклично стойностите на три променливи. Например, ако X=1, Y=2, Z=3, в резултат ще се получи X=2, Y=3, Z=1.

2. Начертайте блок схема на алгоритъм, който при въведени страни на правоъгълника a и b изчислява лицето и обиколката му.

3. Начертайте блок схема на алгоритъм, който при въведени страни на паралелепипед a, b и c изчислява обема му и обема на паралелепипед с два пъти по-малки страни.

4. Съсътавете блок схема на алгоритъма за изчисляване на заплатите, съгласно следните правила:
Ако стажа на работника е по малък от 5 години – заплатата е 500лв.
При стаж от 5 до15 години – 700лв., и при стаж над 15 години заплатата се увеличава с 25 лв. За всяка година.

5.Съсътавете блок схема на алгоритъма за определяне на ученически стипендии, съгласно следните правила:

Ако успеха на ученика е по висок от 5,50 – стипендията е 21лв.
Ако успеха на ученика е по висок от 4,50., и по малък от 5,50 – стипендията е 18лв.
При по нисък успех не се получава стипендия.

6.Голяма фирма търси на работа специалист **ИКОНОМИСТ - ИНФОРМАТИК**, с не по малко от 5 години трудов стаж и не по възрастен от 40 години. Съставете блак схема на алгоритъм, който да определя дали кандидатите ще бъдат приети или не.