Лабораторно упражнение № 4

Тема: Реализиране на линейни програми.

.

I. Цел на лабораторното упражнение

Аритметични и логически операции в програмният език С. Реализиране на линейни програми. Математически функции в програмният език С. Побитови операции.

II. Постановка на задачата

В настоящото упражнение студентите ще се запознаят с интегрираната среда. Ще бъде алгоритмизирани, програмирани, настроени и изпълнени прости линейни програми. Студентите ще се запознаят с побитовите операции и работата с тях.

III. Теоретични сведения

Съществуват три основни вида алгоритми: линейни, разклонени и циклични. При линейни алгоритми действията се изпълняват последователно едно след друго. Всяка стъпка на алгоритъма има само една предходна и една следваща (фиг. 1).

Нека да разгледаме алгоритъма за пресмятане периметъра на правоъгълник по формулата $S = 2 * (a + b)$.

Стъпка 1: Начало.

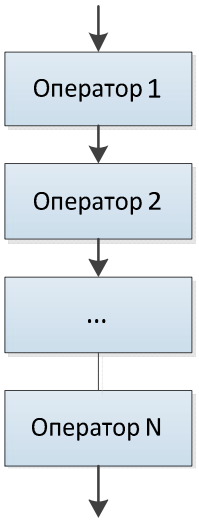
Стъпка 2: Въведете и запомнете стойностите на a и b.

Стъпка 3: На променливата S присвояваме произведението на $2(a + b)$

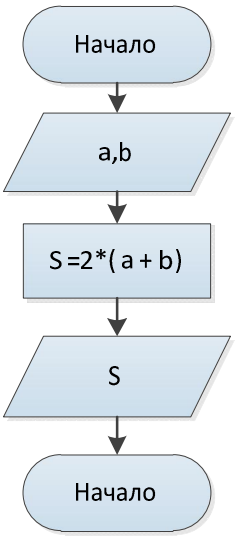
Стъпка 4: Изведете стойността на S като резултат.

Стъпка 5: Край.

Блоковата схема на алгоритъма е дадена на фиг. 2



Фиг. 1. Блокова схема на линеен алгоритъм



Фиг. 2. Блокова схема на алгоритъм за решаване на формулата $S = 2(a + b)$

За да бъде реализирана тази задача е необходимо да бъдат използвани операции както за въвеждане и извеждане на данни, така също и за изпълнение на математически операции.

Програмата ще работи в конзолна среда. Ще покажем две реализации: с класическите входно/изходни операции в езика C и посредством механизма на входно/изходните потоци.

Ето и самото решение:

Програмен код C

```c
#include <stdio.h>
int main(int argc, char *argv[]) {
  float P; // Сума
  float a; // Първо число
  float b; // Второ число
  printf("a =");//Извеждане на текста a=
  scanf("%f",&a); //Въвеждане на a
  printf("b =");//Извеждане на текста b=
  scanf("%f",&b); //Въвеждане на b
  P = 2*(a + b); // Пресмятане на формулата
  printf("2*(%f + %f) =  %f\n",a,b,S);
  return 0;
}
```

Програмен код C++

```c
#include <iostream>
using namespace std;
int main(int argc, char* argv[]) {
  float P; // Сума
  float a; // Първо число
  float b; // Второ число
  cout << "a =";//Извеждане на текста a=
  cin >> a; //Въвеждане на a
  cout << "b ="//Извеждане на текста a=
  cin >> b; //Въвеждане на b
  P = 2*(a + b); // Пресмятане на формулата
  cout <<"2*("<< a <<"+"<<b<<") =" <<S<< endl;
  return 0;

}
```

Математически функции

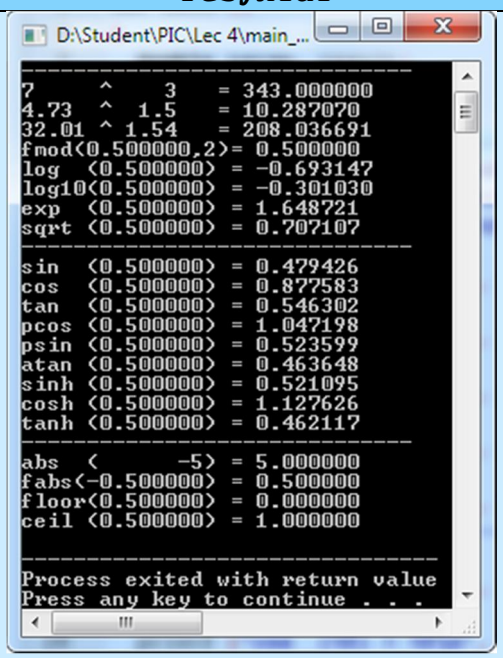
Програмният език C не е специализиран за извършване на математически операции като някой продукти от рода на MatlLab, MatCad, Matematika и др. За да могат да се реализират операции от рода на работа с матрици или решаване на диференциални уравнения е необходимо да се напишат съответните функции. Както и да се използват готови математически библиотеки разпространяване от дадени фирми специализирали се в тази област. От друга страна езикът C разполага с множество елементарни математически функции,на базата на които могат да се създадат разширени библиотеки. Такава е библиотека е стандартната библиотека math.h. Тук ще опишем част от най-често използваните математически функции използвани в C/C++ (Таблица 6.).

Таблица 1. Математически функции

Функция	Предназначение
sin(x)	Синус, sin x, x е в радиани
cos(x)	Косинус, cos x, x е в радиани
tan(x)	Тангенс, tg x, x е в радиани
asin(x)	Аркуссинус, arcsin x ∈ [-π/2, π/2], x ∈ [-1, 1]
acos(x)	Аркускосинус, arccos x ∈ [0, π], x ∈ [-1, 1]
atan(x)	Аркустангенс, arctg x ∈ (-π/2, π/2)
exp(x)	Експонента, e^x
log(x)	Натурален логаритъм, ln x, x > 0
log10(x)	десетичен логаритъм, lg x, x > 0
sinh(x)	хиперболичен синус, sh x
cosh(x)	хиперболичен косинус, ch x
tanh(x)	хиперболичен тангенс, th x
ceil(x)	най-малкото цяло ≥ x, преобразувано в тип double
floor(x)	най-голямото цяло ≤ x, преобразувано в тип double
fabs(x)	абсолютна стойност на x, |x|
sqrt(x)	Корен квадратен от x, x ≥ 0
pow(x, n)	степенуване, x^n (x и n са реални от тип double).

Програмен код	Резултат

```
#include <iostream>
#include <math.h>
int main () {
  double param, result;
  param = 0.5;
  printf("------------------------------\n");
  printf ("7      ^    3    = %f\n", pow (7.0, 3.0) );
  printf ("4.73   ^  1.5    = %f\n", pow (4.73, 1.5) );
  printf ("32.01 ^ 1.54    = %f\n", pow (32.01, 1.54) );
  result = fmod(param,2);
  printf ("fmod(%f,2)= %f\n", param, result );
  result = log  (param);
  printf ("log  (%f) = %f\n", param, result );
  result = log10(param);
  printf ("log10(%f) = %f\n", param, result );
  result = exp  (param);
  printf ("exp  (%f) = %f\n", param, result );
  result = sqrt (param);
  printf ("sqrt (%f) = %f\n", param, result );
  printf("------------------------------\n");
  result = sin  (param);
  printf ("sin  (%f) = %f\n", param, result );
  result = cos  (param);
  printf ("cos  (%f) = %f\n", param, result );
  result = tan  (param);
  printf ("tan  (%f) = %f\n", param, result );
  result = acos (param);
  printf ("àcos (%f) = %f\n", param, result );
  result = asin (param);
  printf ("àsin (%f) = %f\n", param, result );
  result = atan (param);
  printf ("atan (%f) = %f\n", param, result );
   result = sinh (param);
  printf ("sinh (%f) = %f\n", param, result );
  result = cosh (param);
  printf ("cosh (%f) = %f\n", param, result );
  result = tanh (param);
  printf ("tanh (%f) = %f\n", param, result );
  printf("------------------------------\n");
  result = abs (-5);
  printf ("abs  (%8d) = %f\n", -5, result );
  result = fabs (-param);
  printf ("fabs(-%f) = %f\n", param, result );
  result = floor(param);
  printf ("floor(%f) = %f\n", param, result );
  result = ceil (param);
  printf ("ceil (%f) = %f\n", param, result );
  return 0;
}
```



Операции с отделни битове (Поразрядни логически операции)

Езикът C++ притежава и някой качества на асемблерските езици. За разлика от много други езици от високо ниво той разполага и с пълен набор от операции за работа с отделни битове на числените стойности. Тези операции позволяват проверяване, установяване или преместване на отделни битове на стойностите на променливи от тип **int** (**short,long,unsigned**) или тип **char**. Поразредните операции често се използват за създаване на драйверни програми за връзка с външни устройства. Поразредните операции се означават със следните символи.

Таблица 3 — Побитови логически операции

Логическа операция	Предназначение
&	поразредно логическо И(AND)
|	поразредно логическо ИЛИ(OR)
^	поразредно логическо ИЗКЛЮЧВАЩО ИЛИ(XOR,EOR)
~	поразредно логическо ДОПЪЛВАНЕ ДО 1
>>	изместване в дясно (RIGHT SHIFT)
<<	изместване в ляво (LEFT SHIFT)
&=	поразредно логическо И(AND)
|=	поразредно логическо ИЛИ(OR) с присвояване
^=	поразредно логическо ИЗКЛЮЧВАЩО ИЛИ(XOR,EOR) с присвояване
>>=	изместване в дясно (RIGHT SHIFT) с присвояване
<<=	изместване в ляво (LEFT SHIFT) с присвояване

В C++ съществуват логическите операции: `ИЛИ` - `||`; `И` - `&&`. В много от вас ще възникне въпроса, "С какво се отличават операциите: `&` и `&&`; `|` и `||` ?". Отговора може да се получи когато се разбере принципа на работа на поразредните логически операции. Веднага може да се каже, че логическите операции `&&` и `||` се използват само за построяване на логически условия. Тогава как поразрядните логически операции се използват при двоичната аритметика? Въпросът е в това, че тези операции работят с битовете на отделните клетки от паметта. С други думи тези оператори разглеждат операндите си като подредена съвкупност от битове. Всеки бит може да има стойност 0 (false) или 1 (true). Операторите за работа с битове позволяват на програмиста да проверява и инициализира индивидуални битове или битови подмножества.

Операндите на тези оператори трябва да бъдат от целочислен тип. Въпреки че те могат да бъдат със или без знак препоръчва се използването на операнди без знак. Как точно се обработва знаковия бит при тези оператори зависи от реализацията им; програми, които работят с дадена реализация може да не работят с друга. По такъв начин използването на операнд без знак подпомага осигуряването на мобилност на програмата.

Първо ще обясним как работи всеки оператор. После ще дадем примери за използването на всеки от операторите за работа с битове.

Всички оператори, с изключение на оператора ~, са бинарни.

При това операндите могат да бъда зададени дори и в десетичен формат. Тук ще разгледаме всяка една операция поотделно.

Поразредни логически операции И (&) и ИЛИ (|)

Операторите `&` и `|` реализират съответно поразрядна конюнкция и

поразрядна дизюнкция. Таблицата на истинност е показана по-горе.

За по-добро разбиране ще дадем няколко примера с числа. За опростяване на изчисленията ще работим с четири разрядни положителни числа. Първо е показан двоичния код на числата а след това и десетичния.

Поразрядно логическо И

```
  1111 = 15                    1000 = 8
&                            &
  1001 = 9                     1010 = 10
1001 = 9                     1000 = 8
```

Поразрядно логическо ИЛИ

```
  1111  = 15                   1000 = 8
|                            |
  1001 =   9                   1010 = 10
  1111 = 15                    1010 = 10
```

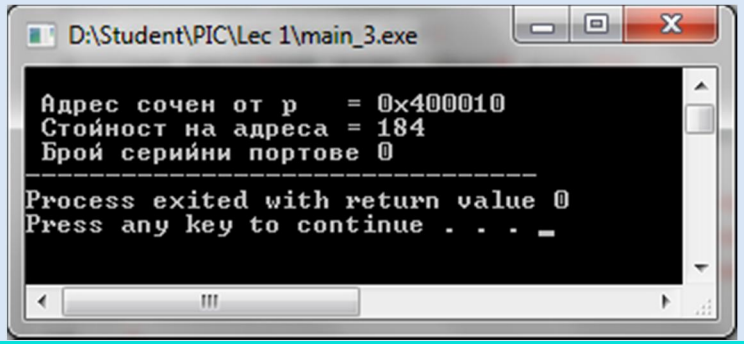
Пример:

На адрес 0040:0010 се съхранява информация за конфигурацията на компютъра. Ако желаем на определим броят на инсталираните серийни канали можем да изпълним следната операция.

Програмен код

```
#include <iostream>
#include <windows.h>
using namespace std;
int main(int argc, char* argv[])
{
  unsigned  int far *p = (unsigned int far *)0x00400010;
  setlocale(LC_ALL,"Bulgarian");
  cout << "\n Адрес сочен от p   = " << p;
  cout << "\n Стойност на адреса = " << *p;
  cout << "\n\t Брой серийни портове " << (( *p & 0x1c00) >> 10);
  return 0;
}
```

Резултат



В тази програма се използва това, че битове 9,100 и 11 съдържат броят на серийните портове. С операцията (*p & 0x1c00) се нулират всички битове с изключение на значещите, а с операцията >> се

ротира информацията в резултата от предходната операция за да може значещата информация да отиде в младшите адреси.

Поразредни логически допълване до 1 (~)

Побитовата операция **"НЕ"** се записва във вида **~операнд**.

Операндът е от цял или символен тип. Поразрядно се извършва операцията отрицание. Операцията е едноместна – всеки бит 0 в операнда става 1, а всеки бит 1 – 0.

Пример:

Програмен код	Резултат

`int i = 0xAA; // 10101010`
`int c;`
`c = ~i;`
	c = 0x55
	или
	c= 01010101

Операция «изключващо ИЛИ» (XOR)

Операцията "**изключващо ИЛИ**" се отличава от операцията **"ИЛИ"** само в една комбинация при която двата оператора приемат стойност 1 (последен ред от таблицата на истинност). Тоест резултатът ще бъде 1 само тогава когато стойностите на двата оператора се различават.

Операцията "**изключващо ИЛИ**" в булевата алгебра се означава със символа "$\oplus$", а в програмният език C/C++ със символа **"^" ("A ^ B")**. Тази операция може да се представи посредством базовите операции ("НЕ", "И", "ИЛИ") със следния израз:

$$A \oplus B = \bar{A}.B + A.\bar{B}$$

Таблица на истинност

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	1

Пример с числа

```
   1111 = 15          1000  = 8
^                  ^
   1001 = 9           1010 = 10
   0110 = 6           0010 = 2
```

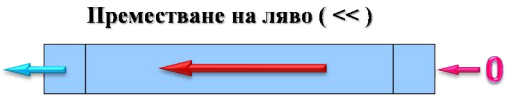
Побитови операции

```
  01101001    01101001    01101001
& 01010101  | 01010101  ^ 01010101   ~ 01010101
  01000001    01111101    00111100     10101010
```

Поразредно логическо преместване на ляво (<<) и преместване на дясно (>>) на разредни битове

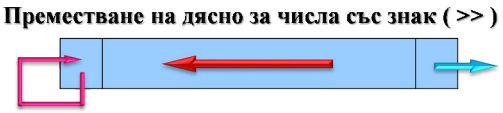
Операторите **<<** и **>>** предизвикват преместване на ляво и съответно преместване надясно на разредите (битовете) на левия операнд с брой позиции, зададени като десен операнд. Например, **x**

<< 2; предизвиква преместване на всички битове на операнда x с две позиции наляво, като освобождаващите се битове се запълват с нули.



Аналогично, **x >> 3;** предизвиква преместване на битовете на x с 3 позиции надясно. Когато операторът **>>** се прилага върху число без знак, освобождаващите се битове се запълват с нули.

Ако числото е със знак, някои компилатори размножават знаковия бит (реализира се аритметично преместване), а други и в този случай запълват освобождаващите се битове с нули. В BORLAND C++ се реализира аритметично преместване надясно.



Преместването наляво с една позиция е равносилно на умножение по две, а преместването надясно с една позиция е равносилно на деление на две . Тези свойства дават възможност за реализиране на умножение и деление по степените на две чрез операторите **<<** и **>>**. Това води до увеличаване на бързодействието, тъй като тези оператори са изключително бързи.

Пример:

```
unsigned char bits = 1; //    0 0 0 0 0 0 0 1
bits = bits << 1;       // -> 0 0 0 0 0 0 1 0
bits = bits << 2;       // -> 0 0 0 0 1 0 0 0
bits = bits >> 3;       // -> 0 0 0 0 0 0 0 1
```

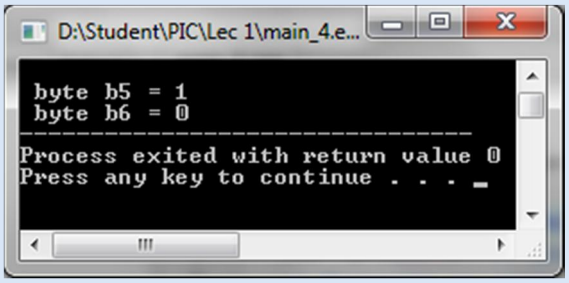
Ето някои характерни приложения на поразредните операции:

```
unsigned x;
//Нулиране на n-тия бит на x
x &= ~(1 << n );
//Установяване в единица на n-тия бит x
x |= 1<< n ;
//Извличане на n-тия бит на x
x & ( 1 << n);
//Извличане на n бита на x, започвайки от p-та позиция
( (x >> (p + 1 - n) ) & ~(~0<< n) );
```

Пример:

Програмен код	Резултат

```
#include <iostream>
using namespace std;
int main(int argc, char* argv[]) {
// Деклариране на цели променливи
// a и b и инициализиране на a
  int a=0xAA,b;
// Идентифициране стойността на бит b5
  b = a & 0x20;
// Преместване 5 позиции на дясно
  b >>= 5;
  cout << "\n byte b5 = " << b;
// Идентифициране стойността на бит b6
 b = a & 0x40;
 b  >>=  6;//  Преместване  6  позиции  на
дясно
  cout << "\n byte b6 = " << b;
  return 0;
}
```



IV. Задание за работа

Зад 1. Да се състави линейна програма на C++, която при въведени страни a, b, c на триъгълник пресмята:

$S = \sqrt{p.(p-a).(p-b).(p-c)}$, където $p = \frac{a+b+c}{2}$, а медианите към страните a, b, c се изчисляват по формулите:

$$m_a = \frac{1}{2}\sqrt{2b^2+2c^2-a^2} \quad ; \quad m_b = \frac{1}{2}.\sqrt{2a^2+2c^2-b^2} \; ; \quad m_c = \frac{1}{2}.\sqrt{2b^2+2a^2-c^2}$$

Зад 2. Да се състави линейна програма на C++, която при въведени параметри R, r, l, h на пресечен конус пресмята:

- пълната повърхнина – $S = \pi(R+r).l + \pi.R^2 + \pi.r^2$
- обемът на пресечения конус – $V = \frac{1}{3}.\pi(R^2+r^2+R.r)h$

Зад 3. Да се състави линейна програма на C++, която при въведен аргумент X пресмята функциите:

$$y_1 = \frac{13{,}5.x^7+9{,}5}{5.(x^2-x-1)} - 4.x^4 \; ; \qquad y_2 = \frac{x^2-3x+2}{\sqrt{2x^2-1}} \; ;$$

$$y_3 = \sqrt{\sin^2(x^3-7{,}2e-2)+\cos^2(x+2{,}53)} \; ; \qquad y_4 = x + \cfrac{x^2}{2.x + \cfrac{x^3}{3.x + \cfrac{x^4}{4}}}$$

Зад 4. Да се състави линейна програма на C++, която при въведени параметри a, b, c, R, H пресмята:

- Обемът на цилиндър – $V_{цил} = S_{окр.}.H; \qquad S_{окр.} = \pi.R^2$

Обемът на вписаната в него триъгълна пирамида – $V_{пир.} = S_{осн.}.H;$

$S = \sqrt{p.(p-a).(p-b).(p-c)}$, където $p = \frac{a+b+c}{2}$

Обемът на частта, която се намира между пирамидата и цилиндъра.

Зад 5. Да се състави *линейна* програма на C++, която при въведен аргумент X пресмята функциите:

$y_1 = \ln(x^4 + \sqrt[3]{x} - e^x + 10)$; $y_2 = \log_5(x^4 + x^2 + 8)$; $y_3 = \sqrt{\frac{y_1}{y_2}}$

Зад 6. Да се състави *линейна* програма на C++, която при въведен аргумент X пресмята функциите:

$y_1 = \sqrt{\frac{\sin(\sin(\sin x)) + \cos(\cos(\cos x))}{|\ln x| + |\cos x| + e^x}}$; $y_2 = \frac{\sin^2(x + e^x)^2}{1 + 3{,}17.\sqrt{x} + e^{\sqrt[3]{x+1}}}$; $y_3 = \frac{y_1.y_2}{y_1 + y_2}$

Зад 7. Да се състави линейна програма на C++, която да пресмята капацитета и съпротивленията на кондензатор по формулите:

- привидно съпротивление - $Z = \frac{U}{I}$
- активно съпротивление - $R = \frac{P}{I^2}$
- капацитет на кондензатора - $C_x = \frac{1}{2\pi.f.\sqrt{Z^2 - R^2}}$

Зад 8. Да се състави линейна програма на C++, която да пресмята стойностите на е.д.н., напрежението U и тока I по формулите:

електродвижещо напрежение $e = E_m.\sin(\frac{2\pi}{T}.t + \psi_e)$

- напрежението U $u = U_m.\sin(\frac{2\pi}{T}.t + \psi_u)$
- тока I $i_m = I_m.\sin(\frac{2\pi}{T}.t + \psi_i)$

Зад 9. Да се състави линейна програма на C++, която да пресмята параметрите на паралелен трептящ кръг по формулите:

$$I_c = 2\pi.f_p.C.U \; ; \quad I_L = \frac{U}{2\pi.f_p.L} \; ; \quad f_k = \frac{1}{2\pi\sqrt{L.C}}$$

Зад 10. Да се състави линейна програма на C++, която да пресмята функцията:

$Ct = \frac{t}{t^2+1} + \sqrt{\frac{t}{t^3+2a}} - e^{-\frac{\lambda t}{a}}$, където $\lambda = \frac{(x-1)^{a-2}}{5+a^2} + \sqrt{\frac{x^3+a}{a^5}}$

Зад 11. Да се състави линейна програма на C++, която да пресмята функцията:

$f(x) = 1 + \sqrt{\frac{x^3-1}{5.x+a^2}} - e^{-\frac{\gamma}{x}}$, където $\gamma = \sqrt{\frac{2\alpha-3}{h^2+1}} - \frac{e^{-\frac{\alpha}{h}}}{3h}$

Зад 12. Да се състави линейна програма на C++, която да пресмята функцията:

$$f(x) = 3{,}5\frac{x-1}{5a^2} + \sqrt{\frac{x^3}{1+a^2}} \quad , \quad \text{където} \quad \alpha = \frac{e^{-x} + \sqrt{(x-1)^5}}{5x^2}$$

Зад 13. Да се състави линейна програма на C++, която да пресмята функцията:

$$\gamma_{VP} = \gamma_0 e^{\frac{VP}{RT}} + \sqrt{\frac{(\Delta T^3 + 1)^3}{2V + P^2}} \text{, където } \Delta T = 1 - \left(\frac{T^2 - 1}{3a + T}\right)^{2a} + \sqrt{\frac{3\alpha - 1}{5T^3}}$$

Зад 14. Каква е разликата между следните оператори?

++i, i++ , i+=1 ,i=i+1 , --i , i-- , i=i-1 , i-=1

j=j*i, j*=i , j=j/i , j/=i

Зад 15. Какъв е резултатът от изпълнението на следната програма.

```
int i=1;
main(){
  int j=20,k;
  double z;
  k=i/j;
  k=i+j;
  k=i-j;
  k=i*j;
  k=i%j;
  z=i/j;
  z = (double)i/(double)j;
}
```

Зад 16. Направете програма която да извежда на екрана съобщението "Hello, world"

Зад 17. Напишете програма която да извършва следните операции:

- Въвеждане на две цели числа от клавиатурата.
- Извеждане на резултата в дясно подравнен формат
- Извеждане на резултата в шестнадесетичен вид.
- Въвеждане на две реални числа и да се изведат на екрана

Зад 18. Напишете програма която да извършва следните операции:

- Да се въведе от клавиатурата следния текст "Това е примерно въвеждане на данни" и да се изведе на екрана.

Зад 19. Напишете програма която да извършва следните операции:

- да се въведат две цели числа
- да се разделят двете числа
- да се определи остатъка и от целочисленото им деление
- резултатът да се изведе на екрана

Зад 20. Напишете програма която да извършва следните операции:

- Да изведе на екрана следният текст "Програмирането е сложно за мързеливия".
- Върху същия текст да се изведе "Програмирането е просто за трудолюбивия".

Зад 21. Напишете програма която да извършва следните операции:

- Да изведе на екрана стойността на 3 бит от числото 0x2F;

- Да се нулира 2 5 и 7 битове от числото 0xAA. Резултата да се изведе на екрана в подходящ формат.

- Да се инвертират 3, 4 и 5 бит от числото 0xAC. Резултата да се изведе на екрана в подходящ формат.

Зад 22. Да се напише програма на C, да се въведе в компютъра и да се реши с нейна помощ контролния пример. Да се визуализира задачата и решението.

- При въвеждане на 3 двойки числа -- координати на върховете на триъгълник да се пресметне лицето му. (2,3); (5,2.5); (0.1,-0.1)
- При въвеждане на 3 двойки числа -- координати на върховете на триъгълник да се пресметне лицето му. (2,3); (5,2.5); (0.1,-0.1)
- При въвеждане на 3 двойки числа -- координати на върховете на триъгълник да се пресметне радиусът на описаната около него окръжност. (2,2.4); (5,2.5); (0.1,-0.1)
- При въвеждане на 3 двойки числа -- координати на върховете на триъгълник да се пресметне отношението на най-малкия му ъгъл към най-големия му ъгъл. (2,2.4); (5,2.5); (0.1,0)
- При въвеждане на 3 двойки числа -- координати на върховете на триъгълник да се пресметне периметъра му. (2,2.4); (5,2.5); (0.1,0)
- При въвеждане на 3 двойки числа -- координати на върховете на триъгълник да се пресметне радиуса на вписаната в триъгълника окръжност. (2,2.1); (-3,2.5); (0,0)
- При въвеждане на 3 двойки числа -- координати на върховете на триъгълник да се пресметнат големините на трите му ъгли. (3,2.1); (-3,2.5); (0,0)
- При въвеждане на 3 двойки числа -- координати на върховете на триъгълник да се пресметне големина на най-големия му ъгъл. (3,4); (-3,2.5); (0,0.2)
- При въвеждане на 3 двойки числа -- координати на върховете на триъгълник да се определи какъв е триъгълника -- остроъгълен, тъпоъгълен или правоъгълен. (0,0); (2,0); (0,5)
- При въвеждане на 3 двойки числа -- координати на върховете на триъгълник да се определи дали триъгълникът е равнобедрен. (3,4); (-1,2.5); (0,0.2)
- При въвеждане на 3 двойки числа -- координати на върховете на триъгълник да се определи дали триъгълникът е правоъгълен. (1,4); (-1,2.5); (0,0.2)
- При въвеждане на 3 числа -- дължините на страните на триъгълник, да се намерят координатите на върховете му, ако едната му страна лежи на абцисната ос, а единият му връх е началото на координатната система.5; 4; 2

- При въвеждане на 3 числа -- коефициентите на уравнението на права „ax+by+c=0”, да се намерят координатите на пресечните точки на правата с координатните оси. 9; -1; 2
- При въвеждане на 3 числа -- коефициентите на уравнението на права „ax+by+c=0”, да се намерят координатите на пресечната точки на правата с друга права с уравнение „x+y=1”. 9; -1; 2
- При въвеждане на 3 числа -- коефициентите на уравнението на права „ax+by+c=0”, да се намерят координатите на пресечните точки на правата с координатните оси. 9; -1; -2
- При въвеждане на 3 числа -- коефициентите на уравнението на права „ax+by+c=0”, да се намерят координатите на пресечната точки на правата с друга права с уравнение „x+y=1”. 3; 1; 2
- При въвеждане на 2 двойки числа -- координати на точки „A” и „C” в равнината, да се намерят координатите на точки „B” и „D” такива, че „ABCD” да е квадрат („AC” да е диагонал на „ABCD”. (3,1); (-1,-1)
- При въвеждане на 2 двойки числа -- координати на точки „A” и „C” в равнината, да се намери лицето на квадрата „ABCD” (с диагонал отсечката „AC”. (-1,-1); (-1,2)
- При въвеждане на 2 двойки числа -- координати на точки „A” и „C” в равнината, да се намерят координатите на точки „B” и „D” такива, че „ABCD” да е квадрат („AC” да е диагонал на „ABCD”. (1,1); (-1,-2)
- При въвеждане на 2 двойки числа -- координати на точки в равнината, да се намери центъра на окръжността, минаваща през тези две точки и през началото на координатната система. (1,1); (-1,2)
- При въвеждане на координати на точка в равнината, да се намерят координатите на върховете на правилен шестоъгълник с център въведената точка и връх началото на координатната система. (1,2)
- При въвеждане на координати на точка в равнината, да се намерят координатите на върховете на правилен шестоъгълник, със зададени два върха -- въведената точка и началото на координатната система. (1,2)
- При въвеждане на координати на точка в равнината и число, да се намерят координатите на върховете на правилен седмоъгълник с център въведената точка и страна въведеното число. (1,1); 1
- При въвеждане на координати на точка в равнината и число, да се намерят координатите на върховете на правилен осмоъгълник с център въведената точка и страна въведеното число. (1,3); 2
- При въвеждане на 3 двойки числа -- координати на върховете на триъгълник в равнината, да се намерят лицата на двата многоъгълника, на които се разделя (евентуално!) триъгълника от ординатната ос. (-3,1); (-1,-1); (2,1)

Зад 23. Да се напише програма която да въвежда едно число от клавиатурата и да изведе побитово инвертираната му

стойност.

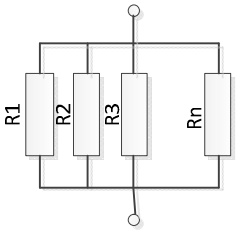
Зад 24. Да се напише програма която да въведе две числа и да определи кои битове съвпадат.

Зад 25. Да се напише програма която да въведе две числа и да определи кои битове не съвпадат.

Зад 26. Да се напише програма която да въведе две числа и да вдигне в единица всички битове от първото число чиито позиции във второто число са 1.

Зад 27. Да се напише програма която да въведе две числа и да вдигне в единица всички битове от първото число чиито позиции във второто число са 0.

Зад 28. Да се напише програма която да намери еквивалентното съпротивление на електрическа верига, съставена от N паралелно свързани резистори по формулата:



$$R = \frac{R_1 R_2 R_3 .... R_n}{R_1 + R_2 + R_3 + ... + R_n}$$

Зад 29. Да се напише програма която да намери еквивалентното съпротивление на електрическа верига, съставена от N последователно свързани резистори по формулата:

$$R = R_1 + R_2 + R_3 + ... + R_n$$

V. Указания за работа

При изпълнението на задачите използвайте примерите приложени в теоретичната част?

За да се използват функциите sin, cos, tan sqrt и др. в началото на програмата поставете кода: #include <math.h>

VI. Контролни въпроси

1. Какви аритметични оператори има в програмният език C?
2. Какви операции за инкементиране познавате?
3. Какви операции за декрементиране познавате?
4. Какви операции за умножение познавате?
5. Какви операции за деление познавате?
6. Как се определя дали едно число е четно или нечетно?
7. Какви функции за въвеждане на данни познавате?
8. Какви функции за извеждане на данни познавате?

9. Посредством коя функция за въвеждане на данни могат да се въведат празни интервали и запетаи?

10. Какви бройни системи се използват в компютърната техника?

11.Как се представят целите числа в компютърната техника?

12.Какво е таблица на истинност?

13. В какъв ред обикновено се записват операндите таблицата на истинност?

14. При какви стойности на операндите A и B изразите "A и B"? "A или B" дават истина?

15. Какви електрически схеми могат да се използват за илюстриране на операциите "И" и "ИЛИ"?

16. Какви знаци се използват за дефиниране на логическите операции "НЕ", "И", "ИЛИ"?

17. Защо операцията "И" се нарича логическо умножение?

18. Защо операцията "ИЛИ" се нарича логическо сумиране?

19. По какво се отличават операциите"изключващо ИЛИ" от "ИЛИ"?

20. Защо операцията "XOR" се нарича сума по модул 2?

21. Защо в таблицата на истинност за операцията НЕ има само два реда, докато при останалите операции има повече?

VII. Литература

1. Кай Хорстман. Принципи на програмирането със C++. ИК Софтех София 2000
2. Симеонов Г. Програмиране на C++. С. Техника 1993г.
3. Тед Фейсон. Borland C++. Обектно-ориентирано програмиране. NISOFT София 1994
4. Богданов Д., И.Мустакеров. Език за програмиране С. С. Техника 1989г.
5. А.Касткин. Профессиональное программирование на языке С. Системное программирование. Минск Вышэйшая школа 1993.