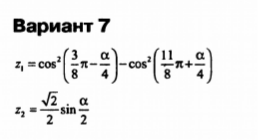
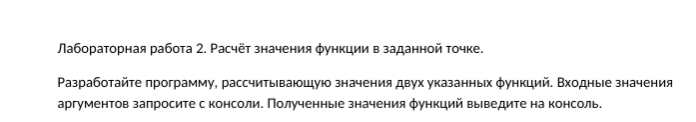
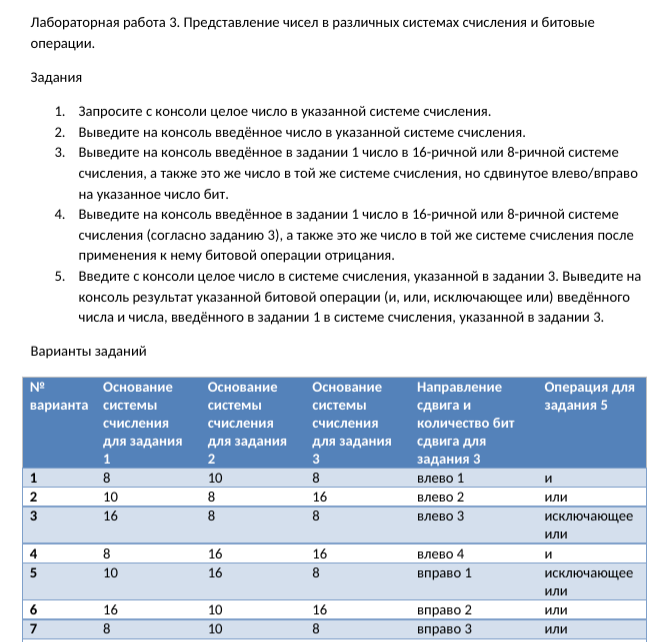
**Лабараторные работы 1-4**

Выполнил студент группы M3105

Клишевич Вадим Александрович

#include <stdio.h>  
  
int main() {  
 char x[256]; //объявляем массив символов, который является строкой  
 scanf("%s", &x); //считываем строку через “%s”  
 printf("String - %s\n", x); //выводим строку  
 char y; //объявляем символ  
 scanf("%c", &y); //считываем символ перевода строки через “%c”  
 scanf("%c", &y); //считываем наш символ  
 printf("Char - %c", y); //выводим символ  
 return 0; //заканчиваем программу  
}

#include <stdio.h>  
#include <math.h> //объявляем библиотеку math.h, в которой есть cos, sin, sqrt  
  
#define PI acos(-1.) // объявляем PI как arccos(-1)  
  
int main() {  
 double x; //объявляем вещественную переменную  
 scanf("%lf", &x); //считываем ее через “%lf”  
 double z1 = cos(3. / 8. \* PI - x / 4.) \* cos(3. / 8. \* PI - x / 4.) -  
 cos(11. / 8. \* PI + x / 4.) \* cos(11. / 8. \* PI + x / 4.);  
 double z2 = sqrt(2.) / 2. \* sin(x / 2.);  
 printf("z1 = %lf, z2 = %lf", z1, z2);  
 return 0;  
}

#include <stdio.h>  
  
int main() {  
 int x; //объявляю переменную типа int  
 scanf("%o", &x); //ввожу восьмеричное число с помощью “%o”  
 printf("%d\n", x); //вывожу в десятичной с помощью “%d”  
 printf("%o %o\n", x, (x >> 3)); //вывожу само число в восьмеричной системе и его же, но сдвинутое вправо на 3 бита с помощью оператора >>  
 printf("%o %o\n", x, ~x); //вывожу само число в восьмеричной, а также число после битового инвертирования  
 int o;  
 scanf("%o", &o); //считываем новую переменную в восьмеричной системе счисления  
 printf("%o\n", o | x); //выводим результат побитового ИЛИ двух переменных в восьмеричной системе  
 return 0;  
}

#include <stdio.h>  
  
int main() {  
  
 long x;  
 scanf("%ld", &x); //cчитываем наше число  
 printf("%ld\n", (x >= 23) && (x <= 32)); //проверям его принадлежность к диапазону [23; 32]  
 long y;  
 scanf("%ld", &y);  
 printf("%ld\n", !!(y & (1 << 6))); //также можно написать (y >> 6) & 1  
 return 0;  
}