Домашнє завдання №5

Скласти програму (C/C++), яка дозволяє знайти невідоме значення x методом повного перебору для заданих констант a, b, c, d, e, f, g та h та порівняти його з обчисленим. x, a, b, c, d, e, f, g та h – дійсні значення одинарної точності(`float`).

Вибір варіанту

(Nж + Nг + 1) % 30 + 1
де: Nж – порядковий номер студента в групі, а Nг – номер групи(1,2,3,4,5,6,7,8 або 9)

Варіанти завдання

Варіант	Вираз
1	x + (a + b) + (c + d) + (e + f) + (g + h) = 0
2	x + (a + b) - (c + d) + (e + f) - (g + h) = 0
3	x + (a + b) * (c + d) + (e + f) * (g + h) = 0
4	x + (a + b) / (c + d) + (e + f) / (g + h) = 0
5	x + (a + b) + (c + d) + (e + f) + (g + h) = 0
6	x + (a + b) - (c + d) + (e + f) - (g + h) = 0
7	x + (a + b) * (c + d) + (e + f) * (g + h) = 0
8	x + (a + b) / (c + d) + (e + f) / (g + h) = 0
9	x + (a + b) + (c + d) + (e - f) + (g - h) = 0
10	x + (a + b) - (c + d) + (e - f) - (g - h) = 0
11	x + (a + b) * (c + d) + (e - f) * (g - h) = 0
12	x + (a + b) / (c + d) + (e - f) / (g - h) = 0
13	x + (a + b) + (c + d) + (e - f) + (g - h) = 0
14	x + (a + b) - (c + d) + (e - f) - (g - h) = 0
15	x + (a + b) * (c + d) + (e - f) * (g - h) = 0
16	x + (a + b) / (c + d) + (e - f) / (g - h) = 0
17	x + (a - b) + (c - d) + (e + f) + (g + h) = 0
18	x + (a - b) - (c - d) + (e + f) - (g + h) = 0
19	x + (a - b) * (c - d) + (e + f) * (g + h) = 0
20	x + (a - b) / (c - d) + (e + f) / (g + h) = 0
21	x + (a - b) + (c - d) + (e + f) + (g + h) = 0
22	x + (a - b) - (c - d) + (e + f) - (g + h) = 0
23	x + (a - b) * (c - d) + (e + f) * (g + h) = 0
24	x + (a - b) / (c - d) + (e + f) / (g + h) = 0
25	x + (a - b) + (c - d) + (e - f) + (g - h) = 0
26	x + (a - b) - (c - d) + (e - f) - (g - h) = 0
27	x + (a - b) * (c - d) + (e - f) * (g - h) = 0
28	x + (a - b) / (c - d) + (e - f) / (g - h) = 0
29	x + (a - b) + (c - d) + (e - f) + (g - h) = 0
30	x + (a - b) - (c - d) + (e - f) - (g + h) = 0

Приклад коду

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>

#define A  0.33333333
#define B  0.
#define C -3.
#define REPEAT_COUNT 1000000
#define REPEATOR(count, code) \
for (unsigned int indexIteration = (count); indexIteration--;){ code; }
#define TWO_VALUES_SELECTOR(variable, firstValue, secondValue) \
        (variable) = indexIteration % 2 ? (firstValue) : (secondValue);

float getCurrentTime(){
        clock_t time = clock();
        if (time != (clock_t)-1) {
                return ((float)time / (float)CLOCKS_PER_SEC);
        }
        return 0.; // else
}
void run_native(float * const dArr){
        float * const dAC = dArr;
        float * const dA = &dAC[0];
        float * const dC = &dAC[1];
        float * const dB = &dArr[2];
        float * const dResult = &dArr[4];
        float * const dX1 = &dResult[1];
        float * const dX2 = &dResult[0];

        REPEATOR(REPEAT_COUNT,
                TWO_VALUES_SELECTOR(*dA, 4., A);
        TWO_VALUES_SELECTOR(*dB, 3., B);
        TWO_VALUES_SELECTOR(*dC, 1., C);
        float vD = sqrt((*dB)*(*dB) - 4.*(*dA)*(*dC));
        (*dX1) = (-(*dB) + vD) / (2.*(*dA));
        (*dX2) = (-(*dB) - vD) / (2.*(*dA));
        )
}
void run_search(float * const dArr){
        float * const dAC = dArr;
        float * const dA = &dAC[0];
        float * const dC = &dAC[1];
        float * const dB = &dArr[2];
        float * const dResult = &dArr[4];
        float * const dX1 = &dResult[1];
        float * const dX2 = &dResult[0];

        *dX1 = 0.; // reset the result
        *dX2 = 0.; // reset the result
        //printf("x1 = %5.2f; x2 = %5.2f;\r\n", *dX1, *dX2);
        *dA = A;
        *dB = B;
        *dC = C;

        unsigned int * const uX1 = (unsigned int * const)dX1;
        for (*uX1 = 0; *uX1 < ~0; ++*uX1){
                if (*dA * *dX1 * *dX1 + *dB * *dX1 + *dC == 0){
                        break;
                }
```

```c
        }

        unsigned int * const uX2 = (unsigned int * const)dX2;
        for (*uX2 = *uX1 + 1; *uX2 < ~0; ++*uX2){
                if (*dA * *dX2 * *dX2 + *dB * *dX2 + *dC == 0){
                        break;
                }
        }
}
void printResult(char * const title, float * const dArr, unsigned int runTime, unsigned int runTimeBySeconds){
        float * const dAC = dArr;
        float * const dA = &dAC[0];
        float * const dC = &dAC[1];
        float * const dB = &dArr[2];
        float * const dResult = &dArr[4];
        float * const dX1 = &dResult[1];
        float * const dX2 = &dResult[0];

        printf("%s:\r\n", title);
        printf("%fx^2 + %fx + %f = 0;\r\n", *dA, *dB, *dC);
        printf("x1 = %5.2f; x2 = %5.2f;\r\n", *dX1, *dX2);
        if (runTime){
                printf("run time: %dns\r\n\r\n", runTime);
        }
        else if (runTimeBySeconds){
                printf("run time: %ds (~%d.000.000.000ns)\r\n\r\n", runTimeBySeconds, runTimeBySeconds);
        }
}
int main() {
        float * const dArr = (float *)malloc(6 * sizeof(float));

        float * const dAC = dArr;
        float * const dA = &dAC[0];
        float * const dC = &dAC[1];
        float * const dB = &dArr[2];
        float * const dResult = &dArr[4];
        float * const dX1 = &dResult[1];
        float * const dX2 = &dResult[0];

        float startTime, endTime;

        // compute by the formula with compiler optimization
        startTime = getCurrentTime();
        run_native(dArr);
        endTime = getCurrentTime();
        printResult((char*)"compute by the formula with compiler optimization",
                dArr,
                (unsigned int)((endTime - startTime) * (1000000000 / REPEAT_COUNT)), 0);

        printf("please wait, the full search takes a few tens of seconds...");
        startTime = getCurrentTime();
        run_search(dArr);
        endTime = getCurrentTime();
        printf("\r                                                    \r");
        printResult((char*)"search",
                dArr,
                0, (unsigned int)(endTime - startTime));

        printf("Press any key to continue . . .");
        getchar();
        return 0;
}
```