Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей
Кафедра информатики
Дисциплина «Конструирование программ»

ОТЧЕТ

к лабораторной работе №8

на тему:

«Интеграция ассемблерных прерываний в проекты на C++»
БГУИР 1-40-04-01

Выполнил студент группы 253504
ЖГУТОВ Евгений Дмитриевич

_____
(дата, подпись студента)

Проверил ассистент кафедры информатики
РОМАНЮК Максим Валерьевич

_____
(дата, подпись преподавателя)

Минск 2023

Цель работы: Задание 1.Вариант 11.Необходимо посчитать корни биквадратного уравнения.

Ход работы: на рисунке 1 – результат работы программы.

Листинг 1 – Исходный код программы задания 1

```
calculate_discriminant proc ; diskriminant
    finit
    fld coeff_a;
    fld coeff_c;
    fmulp ;
    fld four;
    fmulp;
    fld coeff_b;
    fmul st(0), st(0) ;
    fsub st(0), st(1);
    fstp discriminant;

    ret

calculate_discriminant endp

calculate_2_roots proc ;D > 0
    finit ; root 1
    fld discriminant
    fsqrt;
    fldz;
    fld coeff_b;
    fsubp;
    fsub st(0),st(1);
    fld two;
    fld coeff_a;
    fmulp;
    fdiv ;
    fstp root1;
    finit ; root 2
    fld discriminant
    fsqrt;
    fldz;
    fld coeff_b;
    fsubp;
    fadd st(0),st(1);
    fld two;
    fld coeff_a;
    fmulp;
```

```
    fdiv ;
    fstp root2;

    ret

calculate_2_roots endp

calculate_1_root proc ;D = 0
    finit ; root 1
    fld discriminant
    fsqrt;
    fldz;
    fld coeff_b;
    fsubp;
    fsub st(0),st(1);
    fld two;
    fld coeff_a;
    fmulp;
    fdiv ;
    fstp root1;
    fld NaN;
    fstp root2;
    ret

calculate_1_root endp

calculate_0_roots proc ;D < 0
    finit ; root 1
    fld NaN;
    fstp root1;
    fld NaN;
    fstp root2;
    ret

calculate_0_roots endp

calculate_A_exception proc ; A = 0
    finit ; root 1
    fld coeff_c;
    fldz;
    fsubp;
    fld coeff_b;
    fdivp;
    fstp root1;
    fld NaN;
```

```
    fstp root2;
    ret

calculate_A_exception endp

calculate_AB_exception proc ; A = 0, B = 0
    finit ; root 1
    fld coeff_c;
    fldz;
    fcom;
    je infinit_sols;
    fld NaN;
    jmp over_inf;
    infinit_sols: fld inf;
    over_inf: fstp root1;
    fld NaN;
    fstp root2;

    ret

calculate_AB_exception endp

solve_equation proc
   finit; CHECK A = 0
   fldz;
   fld coeff_a;
   fcom;
   fnstsw ax;
   sahf;
   jne nonzero_A;
   fldz; CHECK A = 0, B = 0
   fld coeff_b;
   fcom;
   fnstsw ax;
   sahf;
   je ABexception

    Aexception:
        call calculate_A_exception;
        jmp end_proc;

    ABexception:
        call calculate_AB_exception;
        jmp end_proc;
```

```
    nonzero_A: ;A <> 0
        output_string discriminant_message;
        call calculate_discriminant;
        finit;
        fld zro;
        fld discriminant;
        fcom;
        fnstsw ax;
        sahf;
        jc error ;
        je roots1

    roots2: ; D > 0
        output_string two_roots_message
        call calculate_2_roots;
        jmp end_proc;

    roots1: ; D = 0
        output_string one_root_message
        call calculate_1_root;
        jmp end_proc;

    error:
        output_string zero_roots_message
        call calculate_0_roots;

    end_proc:
        ret

solve_equation endp


handle_biquadratic_roots proc
   ;root1
   finit
   fldz;
   fld root1;
   fcom;
   fnstsw ax;
   sahf;
   jc NoRoots1
   je OneRoot1;
   jnc TwoRoots1;
   jmp Exception1;
```

```
NoRoots1:
     output_string zero_root1_message
     fld NaN;
     fstp b1root1;
     fld NaN;
     fstp b2root1;
     jmp EndRoot1
 OneRoot1:
     push di
     lea di, count
     add [di],1
     pop di
     output_string one_root1_message
     fldz;
     fstp b1root1;
     fld NaN;
     fstp b2root1;
     jmp EndRoot1

 TwoRoots1:
     push di
     lea di, count
     add [di], 2
     pop di
     output_string roots1_root1_message
     fld root1;
     fsqrt;
     fst b1root1;
     fchs;
     fst b2root1;
     jmp EndRoot1

 Exception1:
     push di
     lea di, count
     mov [di], 5
     pop di
     fld root1;
     fld inf;
     fnstsw ax;
     sahf;
     jne NoRoots1;
     finit
     fld inf;
     fld inf;
```

```
    fstp b1root1;
    fstp b2root1 ;

EndRoot1:
    ;root2
    finit
    fldz;
    fld root2;
    fcom;
    fnstsw ax;
    sahf;
    jc NoRoots2
    je OneRoot2;
    jnc TwoRoots2;
    jmp Exception2;

NoRoots2:
    output_string zero_root2_message
    fld NaN;
    fstp b1root2;
    fld NaN;
    fstp b2root2;
    jmp EndRoot2

OneRoot2:
    push di
    lea di, count
    add [di], 1
    pop di
    output_string one_root2_message
    fldz;
    fstp b1root2;
    fld NaN;
    fstp b2root2;
    jmp EndRoot2

TwoRoots2:
    push di
    lea di, count
    add [di], 2
    pop di
    output_string roots2_root2_message
    fld root2;
    fsqrt;
    fst b1root2;
```

```
        fchs;
        fst b2root2;
        jmp EndRoot2

    Exception2:
        push di
        lea di, count
        mov [di], 5
        pop di
        fld root2;
        fld inf;
        fnstsw ax;
        sahf;
        jne NoRoots2;
        finit
        fld inf;
        fld inf;
        fstp b1root2;
        fstp b2root2 ;

    EndRoot2:
        ret

handle_biquadratic_roots endp

_asmsolution proc
    mov ax,@data;
    mov ds,ax;
    mov es,ax;

    ;input a , b , c
    output_string a_input_message
    call input_processing
    finit;
    fld result_value;
    fstp coeff_a;

    output_string b_input_message
    call input_processing
    finit;
    fld result_value;
    fstp coeff_b;

    output_string c_input_message
    call input_processing
```

```
    finit;
    fld result_value;
    fstp coeff_c;

    call solve_equation;
    call handle_biquadratic_roots;

    mov bx,1;

    fld b1root1;
    fld b2root1;
    fld b1root2;
    fld b2root2;
    output_string exit_message;
    ret

_asmsolution  endp
```

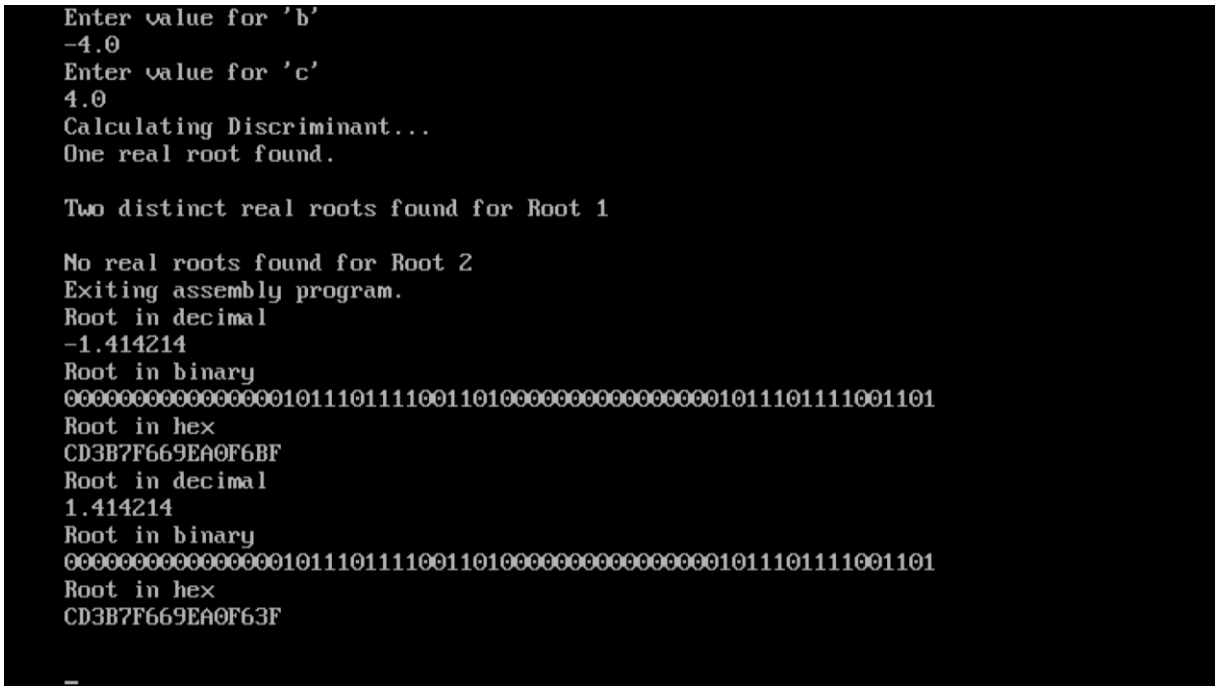

Рисунок 1 – Результат работы программы

Выводы: В результате лабораторной работы была выполнена одна задача с интегрированием ассемблерных прерываний в проект на C++.