МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

НОВОСИБИРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет информационных технологий

Кафедра параллельных вычислений

ОТЧЕТ

О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ

«Введение в архитектуру x86/x86-64 и анализ asm - листингов»

студента 2 курса, 24203 группы

Анисимова Льва Евгеньевича

Направление 09.03.01 – «Информатика и вычислительная техника»

Преподаватель:
доцент, кандидат технических наук
А. Ю. Власенко

Новосибирск 2025

СОДЕРЖАНИЕ

ЦЕЛЬ

Познакомиться с программной архитектурой x86/x86-64 и проанализировать ассемблерные листинги программы для архитектуры x86/x86-64.

ЗАДАНИЕ

Изучить программную архитектуру x86/x86-64:

- набор регистров,
- основные арифметико-логические команды,
- способы адресации памяти,
- способы передачи управления,
- работу со стеком,
- вызов подпрограмм,
- передачу параметров в подпрограммы и возврат результатов,
- работу с арифметическим сопроцессором,
- работу с векторными расширениями.

Для программы алгоритма вычисления числа Пи с помощью разложения в ряд Грегори-Лейбница на языке C++ сгенерировать ассемблерные листинги (синтаксис AT&T, принятый в UNIX) для архитектуры x86 или архитектуры x86-64, используя уровни оптимизации O0 и один из уровней O3/Ofast.

Проанализировать полученные листинги и сделать следующее:

- сопоставить команды языка Си с машинными командами;
- определить размещение переменных языка Си в программах на ассемблере;
- выписать оптимизационные преобразования, выполненные компилятором;

Составить отчет по лабораторной работе.

ОПИСАНИЕ РАБОТЫ

В работе использовалась программа, реализующая алгоритм вычисления числа Пи с помощью разложения в ряд Грегори-Лейбница, написанная для практической работы №1;

С помощью ресурса https://godbolt.org/ были сгенерированы ассемблерные листинги выполнения программы с уровнями оптимизации O0(См. Приложение 2) и O3(См. Приложение 3);

Произведен подробный разбор листингов. На уровне O0 были сопоставлены команды ассемблера и языка C, на уровне O3 - рассмотрены оптимизационные преобразования (См. Приложение 4).

ЗАКЛЮЧЕНИЕ

Изучены программные архитектуры x86 и x86-64. Рассмотрены листинги выполнения программы с разными уровнями оптимизации. На уровне O3 были применены следующие оптимизационные преобразования:

Сокращение количества обращений в память;

Более эффективная обработка циклов;

Уменьшение количества локальных переменных;

Проводятся тестовые запуски подпрограммы, что ускоряет работу всей программы.

Приложение 1. *Код программы*

```
#include <iomanip>
#include <iostream>
#include <sstream>
#include <time.h>

using namespace std;

long double calculate_pi(long long number) {
    long double result = 0;
    int sign = 1;

    for (long long i = 0; i <= number; i++) {
        result += static_cast<long double>(sign) / (2 * i + 1);
        sign = -sign;
    }

    return result;
}

int main(int argc, char** argv)
{
    struct timespec start, end;
    clock_gettime(CLOCK_MONOTONIC_RAW, &start);

    stringstream ss(argv[1]);
    long long int number;

    if (argc != 2) {
        cout << "Usage: " << argv[0] << " <number>" << endl;
        return 1;
    }

    if (ss >> number) {
        cout << "Number: " << number << endl;
    } else {
        cerr << "Invalid number" << endl;
        return 1;
    }

    long double result = calculate_pi(number);

    cout << setprecision(40) << "Pi with your number is about "
<< abs(4 * result) << endl;

    clock_gettime(CLOCK_MONOTONIC_RAW, &end);
    printf("Time taken: %lf sec.\n", end.tv_sec - start.tv_sec +
0.000000001 * (end.tv_nsec - start.tv_nsec));
    return 0;
}
```

Приложение 2. *Ассемблерный листинг с оптимизацией O0*

```
calculate_pi(long long):
        pushq   %rbp
        movq    %rsp, %rbp
        movq    %rdi, -40(%rbp)
        fldz
        fstpt   -16(%rbp)
        movl    $1, -20(%rbp)
        movq    $0, -32(%rbp)
        jmp     .L16
.L17:
        fildl   -20(%rbp)
        movq    -32(%rbp), %rax
        addq    %rax, %rax
        addq    $1, %rax
        movq    %rax, -48(%rbp)
        fildq   -48(%rbp)
        fdivrp  %st, %st(1)
        fldt    -16(%rbp)
        faddp   %st, %st(1)
        fstpt   -16(%rbp)
        negl    -20(%rbp)
        addq    $1, -32(%rbp)
.L16:
        movq    -32(%rbp), %rax
        cmpq    -40(%rbp), %rax
        jle     .L17
        fldt    -16(%rbp)
        popq    %rbp
        ret
.LC2:
        .string "Usage: "
.LC3:
        .string " <number>"
.LC4:
        .string "Number: "
.LC5:
        .string "Invalid number"
.LC6:
        .string "Pi with your number is about "
.LC9:
        .string "Time taken: %lf sec.\n"
main:
        pushq   %rbp
        movq    %rsp, %rbp
        pushq   %rbx
        subq    $536, %rsp
        movl    %edi, -532(%rbp)
        movq    %rsi, -544(%rbp)
        leaq    -96(%rbp), %rax
        movq    %rax, %rsi
        movl    $4, %edi
        call    clock_gettime
        movl    $8, %esi
        movl    $16, %edi
```

```
        call    std::operator|(std::_Ios_Openmode, std::_Ios_Openmode)
        movl    %eax, %ebx
        leaq    -41(%rbp), %rax
        movq    %rax, -40(%rbp)
        nop
        nop
        movq    -544(%rbp), %rax
        addq    $8, %rax
        movq    (%rax), %rcx
        leaq    -41(%rbp), %rdx
        leaq    -80(%rbp), %rax
        movq    %rcx, %rsi
        movq    %rax, %rdi
                        call            std::__cxx11::basic_string<char,
std::char_traits<char>,
std::allocator<char>>::basic_string<std::allocator<char>>(char  const*,
std::allocator<char> const&)
        leaq    -80(%rbp), %rcx
        leaq    -512(%rbp), %rax
        movl    %ebx, %edx
        movq    %rcx, %rsi
        movq    %rax, %rdi
                    call          std::__cxx11::basic_stringstream<char,
std::char_traits<char>,
std::allocator<char>>::basic_stringstream(std::__cxx11::basic_string<ch
ar,      std::char_traits<char>,      std::allocator<char>>      const&,
std::_Ios_Openmode) [complete object constructor]
        leaq    -80(%rbp), %rax
        movq    %rax, %rdi
                        call            std::__cxx11::basic_string<char,
std::char_traits<char>,          std::allocator<char>>::~basic_string()
[complete object destructor]
        leaq    -41(%rbp), %rax
        movq    %rax, %rdi
          call     std::__new_allocator<char>::~__new_allocator()  [base
object destructor]
        nop
        cmpl    $2, -532(%rbp)
        je      .L20
        movl    $.LC2, %esi
        movl    $_ZSt4cout, %edi
            call      std::basic_ostream<char,  std::char_traits<char>>&
std::operator<<<std::char_traits<char>>(std::basic_ostream<char,
std::char_traits<char>>&, char const*)
        movq    %rax, %rdx
        movq    -544(%rbp), %rax
        movq    (%rax), %rax
        movq    %rax, %rsi
        movq    %rdx, %rdi
            call      std::basic_ostream<char,  std::char_traits<char>>&
std::operator<<<std::char_traits<char>>(std::basic_ostream<char,
std::char_traits<char>>&, char const*)
        movl    $.LC3, %esi
        movq    %rax, %rdi
            call      std::basic_ostream<char,  std::char_traits<char>>&
std::operator<<<std::char_traits<char>>(std::basic_ostream<char,
std::char_traits<char>>&, char const*)
                                                          movl
```

```
 $_ZSt4endlIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_, %esi
        movq    %rax, %rdi
                call          std::ostream::operator<<(std::ostream&
(*)(std::ostream&))
        movl    $1, %ebx
        jmp     .L21
.L20:
        leaq    -520(%rbp), %rdx
        leaq    -512(%rbp), %rax
        movq    %rdx, %rsi
        movq    %rax, %rdi
        call    std::istream::operator>>(long long&)
        movq    (%rax), %rdx
        subq    $24, %rdx
        movq    (%rdx), %rdx
        addq    %rdx, %rax
        movq    %rax, %rdi
         call    std::basic_ios<char, std::char_traits<char>>::operator
bool() const
        testb   %al, %al
        je      .L22
        movl    $.LC4, %esi
        movl    $_ZSt4cout, %edi
           call      std::basic_ostream<char,  std::char_traits<char>>&
std::operator<<<std::char_traits<char>>(std::basic_ostream<char,
std::char_traits<char>>&, char const*)
        movq    %rax, %rdx
        movq    -520(%rbp), %rax
        movq    %rax, %rsi
        movq    %rdx, %rdi
        call    std::ostream::operator<<(long long)
                                                        movl
 $_ZSt4endlIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_, %esi
        movq    %rax, %rdi
                call          std::ostream::operator<<(std::ostream&
(*)(std::ostream&))
        jmp     .L31
.L22:
        movl    $.LC5, %esi
        movl    $_ZSt4cerr, %edi
           call      std::basic_ostream<char,  std::char_traits<char>>&
std::operator<<<std::char_traits<char>>(std::basic_ostream<char,
std::char_traits<char>>&, char const*)
                                                        movl
 $_ZSt4endlIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_, %esi
        movq    %rax, %rdi
                call          std::ostream::operator<<(std::ostream&
(*)(std::ostream&))
        movl    $1, %ebx
        jmp     .L21
.L31:
        movq    -520(%rbp), %rax
        movq    %rax, %rdi
        call    calculate_pi(long long)
        fstpt   -32(%rbp)
        movl    $40, %edi
        call    std::setprecision(int)
        movl    %eax, %esi
```

```
        movl    $_ZSt4cout, %edi
            call      std::basic_ostream<char,  std::char_traits<char>>&
std::operator<<<char,  std::char_traits<char>>(std::basic_ostream<char,
std::char_traits<char>>&, std::_Setprecision)
        movl    $.LC6, %esi
        movq    %rax, %rdi
            call      std::basic_ostream<char,  std::char_traits<char>>&
std::operator<<<std::char_traits<char>>(std::basic_ostream<char,
std::char_traits<char>>&, char const*)
        movq    %rax, %rbx
        fldt    -32(%rbp)
        fldt    .LC7(%rip)
        fmulp   %st, %st(1)
        leaq    -16(%rsp), %rsp
        fstpt   (%rsp)
        call    std::abs(long double)
        addq    $16, %rsp
        leaq    -16(%rsp), %rsp
        fstpt   (%rsp)
        movq    %rbx, %rdi
        call    std::ostream::operator<<(long double)
        addq    $16, %rsp
                                                        movl
 $_ZSt4endlIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_, %esi
        movq    %rax, %rdi
                   call          std::ostream::operator<<(std::ostream&
(*)(std::ostream&))
        leaq    -112(%rbp), %rax
        movq    %rax, %rsi
        movl    $4, %edi
        call    clock_gettime
        movq    -112(%rbp), %rdx
        movq    -96(%rbp), %rax
        subq    %rax, %rdx
        pxor    %xmm1, %xmm1
        cvtsi2sdq       %rdx, %xmm1
        movq    -104(%rbp), %rdx
        movq    -88(%rbp), %rax
        subq    %rax, %rdx
        pxor    %xmm2, %xmm2
        cvtsi2sdq       %rdx, %xmm2
        movsd   .LC8(%rip), %xmm0
        mulsd   %xmm2, %xmm0
        addsd   %xmm0, %xmm1
        movq    %xmm1, %rax
        movq    %rax, %xmm0
        movl    $.LC9, %edi
        movl    $1, %eax
        call    printf
        movl    $0, %ebx
.L21:
        leaq    -512(%rbp), %rax
        movq    %rax, %rdi
                   call          std::__cxx11::basic_stringstream<char,
std::char_traits<char>,    std::allocator<char>>::~basic_stringstream()
[complete object destructor]
        movl    %ebx, %eax
        jmp     .L32
```

```
        movq    %rax, %rbx
        leaq    -80(%rbp), %rax
        movq    %rax, %rdi
                        call            std::__cxx11::basic_string<char,
std::char_traits<char>,          std::allocator<char>>::~basic_string()
[complete object destructor]
        jmp     .L26
        movq    %rax, %rbx
.L26:
        leaq    -41(%rbp), %rax
        movq    %rax, %rdi
          call     std::__new_allocator<char>::~__new_allocator()  [base
object destructor]
        nop
        movq    %rbx, %rax
        movq    %rax, %rdi
        call    _Unwind_Resume
        movq    %rax, %rbx
        leaq    -512(%rbp), %rax
        movq    %rax, %rdi
                    call          std::__cxx11::basic_stringstream<char,
std::char_traits<char>,    std::allocator<char>>::~basic_stringstream()
[complete object destructor]
        movq    %rbx, %rax
        movq    %rax, %rdi
        call    _Unwind_Resume
.L32:
        movq    -8(%rbp), %rbx
        leave
        ret
                        .set            std::__cxx11::basic_string<char,
std::char_traits<char>,
std::allocator<char>>::_Alloc_hider::~_Alloc_hider()  [complete  object
destructor],_ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEE12_All
oc_hiderD2Ev
                        .set            std::__cxx11::basic_string<char,
std::char_traits<char>,          std::allocator<char>>::~basic_string()
[complete                                                       object
destructor],_ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEED2Ev
.LC10:
        .string "basic_string: construction from null is not valid"
                        .set            std::__cxx11::basic_string<char,
std::char_traits<char>,
std::allocator<char>>::basic_string<std::allocator<char>>(char  const*,
std::allocator<char>
const&),_ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEC2IS3_EEPK
cRKS3_
                .set       std::__new_allocator<char>::~__new_allocator()
[complete                                                       object
destructor],std::__new_allocator<char>::~__new_allocator() [base object
destructor]
                        .set            std::__cxx11::basic_string<char,
std::char_traits<char>,
std::allocator<char>>::_Alloc_hider::_Alloc_hider(char*,
std::allocator<char>          const&)          [complete          object
constructor],_ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEE12_Al
loc_hiderC2EPcRKS3_
                    .set         void    std::__cxx11::basic_string<char,
```

```
std::char_traits<char>,        std::allocator<char>>::_M_construct<char
const*>(char              const*,              char              const*,
std::forward_iterator_tag)::_Guard::_Guard(std::__cxx11::basic_string<c
har, std::char_traits<char>, std::allocator<char>>*) [complete object
constructor],_ZZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEE12_M
_constructIPKcEEvT_S8_St20forward_iterator_tagEN6_GuardC2EPS4_
                  .set         void    std::__cxx11::basic_string<char,
std::char_traits<char>,        std::allocator<char>>::_M_construct<char
const*>(char              const*,              char              const*,
std::forward_iterator_tag)::_Guard::~_Guard()      [complete      object
destructor],_ZZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEE12_M_
constructIPKcEEvT_S8_St20forward_iterator_tagEN6_GuardD2Ev
.LC11:
        .string "basic_string::_M_create"
.LC7:
        .long   0
        .long   -2147483648
        .long   16385
        .long   0
.LC8:
        .long   -400107883
        .long   1041313291
```

Приложение 3. *Ассемблерный листинг с оптимизацией O3*

```
std::basic_ostream<char, std::char_traits<char>>& std::endl<char,
std::char_traits<char>>(std::basic_ostream<char, std::char_traits<char>>&)
(.isra.0.cold):
calculate_pi(long long):
        testq   %rdi, %rdi
        js      .L14
        leaq    3(%rdi,%rdi), %rcx
        fldz
        movl    $1, %eax
        movl    $1, %edx
.L13:
        movl    %edx, -16(%rsp)
        fildl   -16(%rsp)
        negl    %edx
        movq    %rax, -16(%rsp)
        addq    $2, %rax
        fildq   -16(%rsp)
        fdivrp  %st, %st(1)
        faddp   %st, %st(1)
        cmpq    %rax, %rcx
        jne     .L13
        ret
.L14:
        fldz
        ret
.LC4:
        .string "basic_string: construction from null is not valid"
.LC5:
        .string "Usage: "
.LC6:
        .string " <number>"
.LC7:
        .string "Number: "
.LC8:
        .string "Invalid number"
.LC9:
        .string "Pi with your number is about "
.LC12:
        .string "Time taken: %lf sec.\n"
main:
        pushq   %rbp
        movq    %rsp, %rbp
        pushq   %r15
        movl    %edi, %r15d
        movl    $4, %edi
        pushq   %r14
        movq    %rsi, %r14
        leaq    -496(%rbp), %rsi
        pushq   %r13
        leaq    -464(%rbp), %r13
        pushq   %r12
        pushq   %rbx
        subq    $488, %rsp
        call    clock_gettime
        movq    8(%r14), %r12
        movq    %r13, -480(%rbp)
        testq   %r12, %r12
        je      .L38
        movq    %r12, %rdi
        call    strlen
        movq    %rax, -528(%rbp)
```

```
        movq    %rax, -448(%rbp)
        cmpq    $15, %rax
        ja      .L42
        cmpq    $1, -528(%rbp)
        jne     .L20
        movzbl  (%r12), %eax
        leaq    -448(%rbp), %rbx
        movb    %al, -464(%rbp)
.L21:
        movq    -448(%rbp), %rax
        movq    -480(%rbp), %rdx
        leaq    -480(%rbp), %rsi
        movq    %rbx, %rdi
        movq    %rax, -472(%rbp)
        movb    $0, (%rdx,%rax)
        movl    $24, %edx
        call    std::__cxx11::basic_stringstream<char,
std::char_traits<char>,
std::allocator<char>>::basic_stringstream(std::__cxx11::basic_string<char,
std::char_traits<char>, std::allocator<char>> const&, std::_Ios_Openmode)
[complete object constructor]
        leaq    -480(%rbp), %rdi
        call    std::__cxx11::basic_string<char, std::char_traits<char>,
std::allocator<char>>::_M_dispose()
        cmpl    $2, %r15d
        je      .L22
        movl    $7, %edx
        movl    $.LC5, %esi
        movl    $std::cout, %edi
        call    std::basic_ostream<char, std::char_traits<char>>&
std::__ostream_insert<char, std::char_traits<char>>(std::basic_ostream<char,
std::char_traits<char>>&, char const*, long)
        movq    (%r14), %rsi
        movl    $std::cout, %edi
        call    std::basic_ostream<char, std::char_traits<char>>&
std::operator<<<std::char_traits<char>>(std::basic_ostream<char,
std::char_traits<char>>&, char const*)
        movl    $9, %edx
        movl    $.LC6, %esi
        movq    %rax, %rdi
        movq    %rax, %r15
        call    std::basic_ostream<char, std::char_traits<char>>&
std::__ostream_insert<char, std::char_traits<char>>(std::basic_ostream<char,
std::char_traits<char>>&, char const*, long)
        movq    %r15, %rdi
        call    std::basic_ostream<char, std::char_traits<char>>&
std::endl<char, std::char_traits<char>>(std::basic_ostream<char,
std::char_traits<char>>&) (.isra.0)
.L26:
        movl    $1, %r14d
.L23:
        movq    %rbx, %rdi
        call    std::__cxx11::basic_stringstream<char,
std::char_traits<char>, std::allocator<char>>::~basic_stringstream()
[complete object destructor]
        leaq    -40(%rbp), %rsp
        movl    %r14d, %eax
        popq    %rbx
        popq    %r12
        popq    %r13
        popq    %r14
        popq    %r15
        popq    %rbp
        ret
```

```
.L20:
        leaq    -448(%rbp), %rbx
        cmpq    $0, -528(%rbp)
        je      .L21
        movq    %r13, %rdi
        jmp     .L19
.L22:
        leaq    -504(%rbp), %rsi
        movq    %rbx, %rdi
        call    std::istream& std::istream::_M_extract<long long>(long long&)
        movq    (%rax), %rdx
        movq    -24(%rdx), %rdx
        movl    32(%rax,%rdx), %eax
        andl    $5, %eax
        movl    %eax, %r14d
        jne     .L24
        movl    $8, %edx
        movl    $.LC7, %esi
        movl    $std::cout, %edi
        call    std::basic_ostream<char, std::char_traits<char>>&
std::__ostream_insert<char, std::char_traits<char>>(std::basic_ostream<char,
std::char_traits<char>>&, char const*, long)
        movq    -504(%rbp), %rsi
        movl    $std::cout, %edi
        call    std::ostream& std::ostream::_M_insert<long long>(long long)
        movq    %rax, %rdi
        call    std::basic_ostream<char, std::char_traits<char>>&
std::endl<char, std::char_traits<char>>(std::basic_ostream<char,
std::char_traits<char>>&) (.isra.0)
        movq    -504(%rbp), %rcx
        testq   %rcx, %rcx
        js      .L31
        leaq    3(%rcx,%rcx), %rcx
        movl    $1, %eax
        movl    $1, %edx
        fldz
.L28:
        movl    %edx, -528(%rbp)
        fildl   -528(%rbp)
        negl    %edx
        movq    %rax, -528(%rbp)
        addq    $2, %rax
        fildq   -528(%rbp)
        fdivrp  %st, %st(1)
        faddp   %st, %st(1)
        cmpq    %rax, %rcx
        jne     .L28
.L27:
        movl    $29, %edx
        movl    $.LC9, %esi
        movl    $std::cout, %edi
        movq    std::cout(%rip), %rax
        fstpt   -528(%rbp)
        movq    -24(%rax), %rax
        movq    $40, std::cout+8(%rax)
        call    std::basic_ostream<char, std::char_traits<char>>&
std::__ostream_insert<char, std::char_traits<char>>(std::basic_ostream<char,
std::char_traits<char>>&, char const*, long)
        fldt    -528(%rbp)
        fmuls   .LC10(%rip)
        pushq   %rsi
        movl    $std::cout, %edi
        pushq   %rsi
        fabs
```

```
        fstpt   (%rsp)
        call    std::ostream& std::ostream::_M_insert<long double>(long
double)
        popq    %rdx
        movq    %rax, %rdi
        popq    %rcx
        call    std::basic_ostream<char, std::char_traits<char>>&
std::endl<char, std::char_traits<char>>(std::basic_ostream<char,
std::char_traits<char>>&) (.isra.0)
        movl    $4, %edi
        leaq    -480(%rbp), %rsi
        call    clock_gettime
        movq    -472(%rbp), %rax
        pxor    %xmm0, %xmm0
        subq    -488(%rbp), %rax
        cvtsi2sdq       %rax, %xmm0
        pxor    %xmm1, %xmm1
        movl    $.LC12, %edi
        movq    -480(%rbp), %rax
        mulsd   .LC11(%rip), %xmm0
        subq    -496(%rbp), %rax
        cvtsi2sdq       %rax, %xmm1
        movl    $1, %eax
        addsd   %xmm1, %xmm0
        call    printf
        jmp     .L23
.L42:
        leaq    -448(%rbp), %rbx
        leaq    -480(%rbp), %rdi
        xorl    %edx, %edx
        movq    %rbx, %rsi
        call    std::__cxx11::basic_string<char, std::char_traits<char>,
std::allocator<char>>::_M_create(unsigned long&, unsigned long)
        movq    %rax, -480(%rbp)
        movq    %rax, %rdi
        movq    -448(%rbp), %rax
        movq    %rax, -464(%rbp)
.L19:
        movq    -528(%rbp), %rdx
        movq    %r12, %rsi
        call    memcpy
        jmp     .L21
.L24:
        movl    $14, %edx
        movl    $.LC8, %esi
        movl    $_ZSt4cerr, %edi
        call    std::basic_ostream<char, std::char_traits<char>>&
std::__ostream_insert<char, std::char_traits<char>>(std::basic_ostream<char,
std::char_traits<char>>&, char const*, long)
        movl    $_ZSt4cerr, %edi
        call    std::basic_ostream<char, std::char_traits<char>>&
std::endl<char, std::char_traits<char>>(std::basic_ostream<char,
std::char_traits<char>>&) (.isra.0)
        jmp     .L26
.L31:
        fldz
        jmp     .L27
        movq    %rax, %rbx
        jmp     .L29
        movq    %rax, %r15
        jmp     .L30
main.cold:
.L38:
        movl    $.LC4, %edi
```

```
        call    std::__throw_logic_error(char const*)
.L29:
        leaq    -480(%rbp), %rdi
        call    std::__cxx11::basic_string<char, std::char_traits<char>,
std::allocator<char>>::_M_dispose()
        movq    %rbx, %rdi
        call    _Unwind_Resume
.L30:
        movq    %rbx, %rdi
        call    std::__cxx11::basic_stringstream<char,
std::char_traits<char>, std::allocator<char>>::~basic_stringstream()
[complete object destructor]
        movq    %r15, %rdi
        call    _Unwind_Resume
.LC10:
        .long   1082130432
.LC11:
        .long   -400107883
        .long   1041313291
```

Приложение 4. *Сравнение*

Функция:

```
calculate_pi(long long):
        pushq   %rbp
        movq    %rsp, %rbp
        movq    %rdi, -40(%rbp)
//number → память
        fldz
        fstpt   -16(%rbp)
//result → память
        movl    $1, -20(%rbp)
//sign → память
        movq    $0, -32(%rbp)
//i → память
        jmp     .L16
.L17:
        fildl   -20(%rbp)
//ЗАГРУЗКА sign ИЗ ПАМЯТИ
        movq    -32(%rbp), %rax
//ЗАГРУЗКА i ИЗ ПАМЯТИ
        addq    %rax, %rax          //2
* i
        addq    $1, %rax            //2
* i + 1
        movq    %rax, -48(%rbp)
//сохранение во временную
        fildq   -48(%rbp)
//загрузка из памяти
        fdivrp  %st, %st(1)
//деление
        fldt    -16(%rbp)
//ЗАГРУЗКА result ИЗ ПАМЯТИ
        faddp   %st, %st(1)
//сложение
        fstpt   -16(%rbp)
//СОХРАНЕНИЕ result В ПАМЯТЬ
        negl    -20(%rbp)
//ИЗМЕНЕНИЕ sign В ПАМЯТИ
        addq    $1, -32(%rbp)
//ИНКРЕМЕНТ i В ПАМЯТИ
.L16:
        movq    -32(%rbp), %rax
//загрузка i
        cmpq    -40(%rbp), %rax
//сравнение с number
        jle     .L17
        fldt    -16(%rbp)
        popq    %rbp
        ret
```

```
calculate_pi(long long):
        testq   %rdi, %rdi
//проверка number >= 0
        js      .L14
//если отрицательный → return 0
        leaq    3(%rdi,%rdi), %rcx
//ПРЕДВЫЧИСЛЕНИЕ: rcx = 2*number + 3
        fldz
//result = 0.0 (В СТЕКЕ FPU!)
        movl    $1, %eax
//eax = 1 (начальное 2*i+1)
        movl    $1, %edx
//edx = 1 (sign В РЕГИСТРЕ!)
.L13:
        movl    %edx, -16(%rsp)
//ВРЕМЕННОЕ сохранение sign
        fildl   -16(%rsp)
//загрузка знака
        negl    %edx
//ИЗМЕНЕНИЕ sign В РЕГИСТРЕ!
        movq    %rax, -16(%rsp)
//временное сохранение
        addq    $2, %rax
//ПРЕДВЫЧИСЛЕНИЕ след. 2*i+1
        fildq   -16(%rsp)
//загрузка (2*i+1)
        fdivrp  %st, %st(1)
//sign / (2*i+1)
        faddp   %st, %st(1)
//result += ... (В СТЕКЕ FPU!)
        cmpq    %rax, %rcx
//сравнение с ПРЕДВЫЧИСЛЕННОЙ границей
        jne     .L13
        ret
.L14:
        fldz
        ret
```

Уменьшен стек:

```
subq    $536, %rsp
```

```
subq    $488, %rsp    # на 48 байт
меньше!
```

Хранение в регистрах:

```
movl    %edi, -532(%rbp)    # argc
movq    %rsi, -544(%rbp)    # argv
```

```
movl    %edi, %r15d         # argc
movq    %rsi, %r14          # argv
```

Инлайнинг setprecision:
 	
`movl $40, %edi`
`call std::setprecision`
	`movq std::cout(%rip), %rax`
	`movq $40, std::cout+8(%rax) #`
	`прямое изменение`

Дробление кода и вычленение частей:
 	
`main:`
`// весь код в одной секции`
`testq %r12, %r12`
`je .L32`
`//обработка ошибки В ТОМ ЖЕ СЕГМЕНТЕ`
`// ...`
`.L32:`
`movl $.LC2, %edi`
`call std::__throw_logic_error`
	`main:`
	`//ТОЛЬКО горячий код`
	`testq %r12, %r12`
	`je .L32`
	`//переход в ХОЛОДНУЮ СЕКЦИЮ`
	`//...`
	
	`main.cold:`
	`//ОТДЕЛЬНАЯ СЕКЦИЯ`
	`.L32:`
	`movl $.LC2, %edi //холодный`
	`код (обработка ошибок)`
	`call std::__throw_logic_error`