Лабараторная работа 1.
Вариант 1
Задание 1.

```
int main() {
     int x = 2;

     int y1 = x * 2;
     int y2 = x + 17;

     int y3;
     if (x >= 0) {
         y3 = 13;
     } else {
         y3 = 7;
     }

     return 0;
 }
```

Задание 2.

g++ 1.cpp -S -O0 1.s

```
int x = 2;
movl    $2, -16(%rbp)

int y1 = x * 2;
movl    -16(%rbp), %eax
addl    %eax, %eax
movl    %eax, -12(%rbp)

int y2 = x + 17;
movl    -16(%rbp), %eax
addl    $17, %eax
movl    %eax, -8(%rbp)

int y3;
if (x >= 0) {
    y3 = 13;
} else {
     y3 = 7;
}
cmpl $0, -16(%rbp)
js .L2
movl $13, -4(%rbp)
jmp .L3

.L2:
     movl $7, -4(%rbp)
.L3:
     movl $0, %eax
     popq %rbp
```

Задание 3

```
template <typename T>
T f(T x) {
     return x * 2;
}

char c = 5;
short s = 5;
long l = 5;
long long ll = 5;
long double ld = 5;

int main() {
    char yc = f(c);
    char ys = f(s);
    char yl = f(l);
    char yll = f(ll);
    char yld = f(ld);

    return 0;
}
```

```
main:
.LFB1:
.cfi_startproc
endbr64
pushq %rbp
.cfi_def_cfa_offset 16
.cfi_offset 6, -16
movq %rsp, %rbp
.cfi_def_cfa_register 6
subq $32, %rsp
movzbl c(%rip), %eax
movsbl %al, %eax
movl %eax, %edi
call _Z1fIcET_S0_
movb %al, -5(%rbp)
```

```
movzwl s(%rip), %eax
cwtl
movl %eax, %edi
call _Z1fIsET_S0_
movb %al, -4(%rbp)
movq (%rip), %rax
movq %rax, %rdi
call _Z1fIlET_S0_
movb %al, -3(%rbp)
movq ll(%rip), %rax
movq %rax, %rdi
call _Z1fIxET_S0_
movb %al, -2(%rbp)
fldt ld(%rip)
leaq -16(%rsp), %rsp
fstpt (%rsp)
call _Z1fIeET_S0_
addq $16, %rsp
fnstcw -18(%rbp)
movzwl -18(%rbp), %eax
orb $12, %ah
movw %ax, -20(%rbp)
fldcw -20(%rbp)
fistps -22(%rbp)
fldcw -18(%rbp)
movzwl -22(%rbp), %eax
movb %al, -1(%rbp)
movl $0, %eax
leave
.cfi_def_cfa 7, 8
ret
.cfi_endproc
```

Char:

```
_Z1fIcET_S0_:

endbr64
pushq %rbp
.cfi_def_cfa_offset 16
.cfi_offset 6, -16
movq %rsp, %rbp
.cfi_def_cfa_register 6
movl %edi, %eax
movb %al, -4(%rbp)
movzbl -4(%rbp), %eax
addl %eax, %eax
popq %rbp
.cfi_def_cfa 7, 8
ret
.cfi_endproc
```

Short:

```
_Z1fIsET_S0_:
.LFB3:
.cfi_startproc
endbr64
pushq %rbp
.cfi_def_cfa_offset 16
.cfi_offset 6, -16
movq %rsp, %rbp
.cfi_def_cfa_register 6
movl %edi, %eax
movw %ax, -4(%rbp)
movzwl -4(%rbp), %eax
addl %eax, %eax
popq %rbp
```

```
.cfi_def_cfa 7, 8
ret
.cfi_endproc
```

long:

```
_Z1fIlET_S0_:
.LFB4:
.cfi_startproc
endbr64
pushq %rbp
.cfi_def_cfa_offset 16
.cfi_offset 6, -16
movq %rsp, %rbp
.cfi_def_cfa_register 6
movq %rdi, -8(%rbp)
movq -8(%rbp), %rax
addq %rax, %rax
popq %rbp
.cfi_def_cfa 7, 8
ret
.cfi_endproc
```

long long:

```
_Z1fIxET_S0_:
.LFB5:
.cfi_startproc
endbr64
pushq %rbp
.cfi_def_cfa_offset 16
.cfi_offset 6, -16
movq %rsp, %rbp
.cfi_def_cfa_register 6
movq %rdi, -8(%rbp)
```

```
movq -8(%rbp), %rax
addq %rax, %rax
popq %rbp
.cfi_def_cfa 7, 8
ret
.cfi_endproc
```

long double:

```
_Z1fIeET_S0_:
.LFB6:
.cfi_startproc
endbr64
pushq %rbp
.cfi_def_cfa_offset 16
.cfi_offset 6, -16
movq %rsp, %rbp
.cfi_def_cfa_register 6
fldt 16(%rbp)
fadd %st(0), %st
popq %rbp
.cfi_def_cfa 7, 8
ret
.cfi_endproc
```

Задание 4

```c
int f(int x) {
    return x * 2;
}

int main() {
    int x = 5;
    int y = f(x);

    return 0;
}
```

```
main:

.LFB1:
.cfi_startproc
endbr64
pushq %rbp
.cfi_def_cfa_offset 16
.cfi_offset 6, -16
movq %rsp, %rbp
.cfi_def_cfa_register 6
subq $16, %rsp
movl $5, -8(%rbp)
movl -8(%rbp), %eax
movl %eax, %edi
call _Z1fi
movl %eax, -4(%rbp)
movl $0, %eax
leave
.cfi_def_cfa 7, 8
ret
.cfi_endproc
```

```
int f(int x)
```

```
.LFB0:
.cfi_startproc
endbr64
pushq %rbp
.cfi_def_cfa_offset 16
.cfi_offset 6, -16
movq %rsp, %rbp
.cfi_def_cfa_register 6
movl %edi, -4(%rbp)
movl -4(%rbp), %eax
addl %eax, %eax
popq %rbp
.cfi_def_cfa 7, 8
ret
.cfi_endproc
```

Задание 5

```
float f(float x) {
    return x * 2;
}

int main() {
    float x = 5.f;
    float y = f(x);

    return 0;
}
```

```
main:
.LFB1:
.cfi_startproc
 endbr64
 pushq   %rbp
 .cfi_def_cfa_offset 16
 .cfi_offset 6, -16
 movq    %rsp, %rbp
 .cfi_def_cfa_register 6
 subq    $16, %rsp
 movss   .LC0(%rip), %xmm0
 movss   %xmm0, -8(%rbp)
 movl    -8(%rbp), %eax
 movd    %eax, %xmm0
 call    _Z1ff
 movd    %xmm0, %eax
 movl    %eax, -4(%rbp)
 movl    $0, %eax
 leave
 .cfi_def_cfa 7, 8
 ret
 .cfi_endproc
```

```
_Z1ff:

.LFB0:
.cfi_startproc
endbr64
pushq   %rbp
.cfi_def_cfa_offset 16
.cfi_offset 6, -16
movq    %rsp, %rbp
.cfi_def_cfa_register 6
movss   %xmm0, -4(%rbp)
movss   -4(%rbp), %xmm0
addss   %xmm0, %xmm0
popq    %rbp
.cfi_def_cfa 7, 8
ret
.cfi_endproc
```

Задание 5

```c
static int  x = 2;

int f() {
    return x * 2;
}

int main() {
    f();

    return 0;
}
```

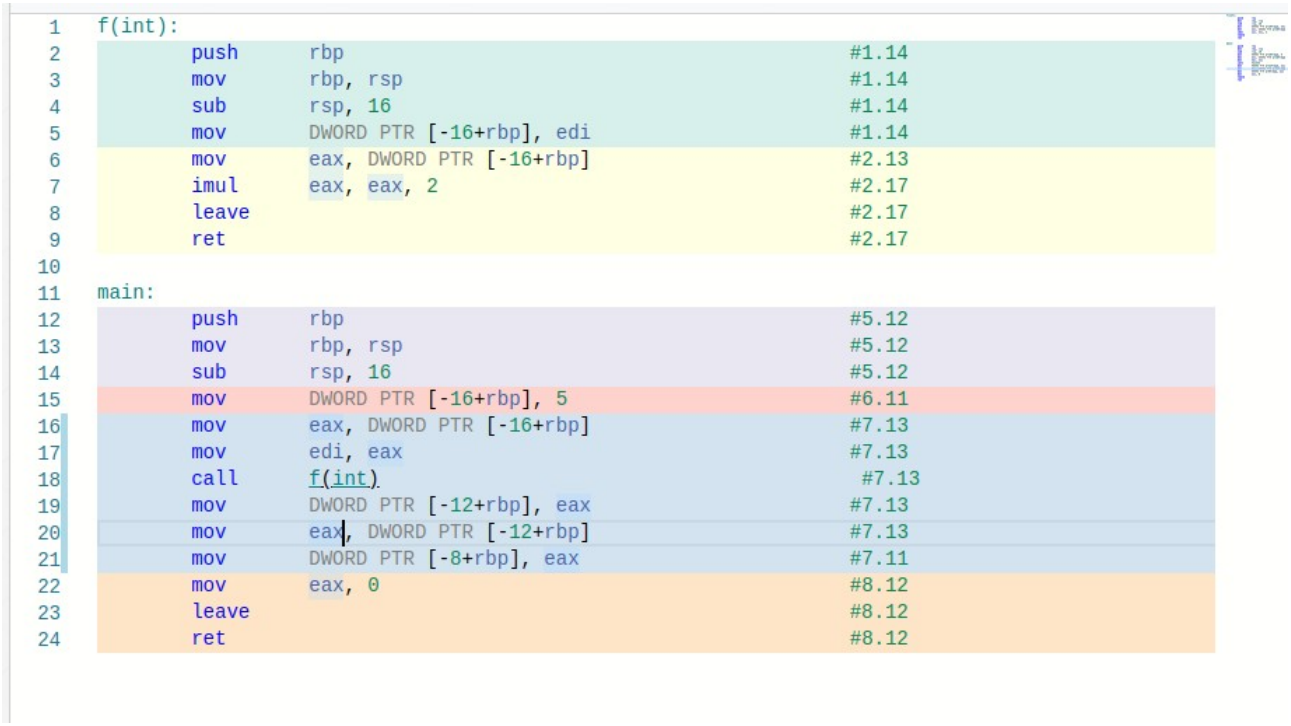
```
_ZL1x:
.long   2
.text
.globl  _Z1fv
.type   _Z1fv, @function

_Z1fv:
.LFB0:
.cfi_startproc
endbr64
pushq   %rbp
.cfi_def_cfa_offset 16
.cfi_offset 6, -16
movq    %rsp, %rbp
.cfi_def_cfa_register 6
movl    _ZL1x(%rip), %eax
addl    %eax, %eax
popq    %rbp
.cfi_def_cfa 7, 8
ret
.cfi_endproc
.LFE0:
.size   _Z1fv, .-_Z1fv
.globl  main
.type   main, @function
```

Задание 6

```
int f(int x) {
     return x * 2;
}

int main() {
     int x = 5.f;
     int y = f(x);
     return 0;
}
```



```
1   f(int):
2           push    rbp                                     #1.14
3           mov     rbp, rsp                                #1.14
4           sub     rsp, 16                                 #1.14
5           mov     DWORD PTR [-16+rbp], edi                #1.14
6           mov     eax, DWORD PTR [-16+rbp]                #2.13
7           imul    eax, eax, 2                             #2.17
8           leave                                           #2.17
9           ret                                             #2.17
10
11  main:
12          push    rbp                                     #5.12
13          mov     rbp, rsp                                #5.12
14          sub     rsp, 16                                 #5.12
15          mov     DWORD PTR [-16+rbp], 5                  #6.11
16          mov     eax, DWORD PTR [-16+rbp]                #7.13
17          mov     edi, eax                                #7.13
18          call    f(int)                                   #7.13
19          mov     DWORD PTR [-12+rbp], eax                #7.13
20          mov     eax, DWORD PTR [-12+rbp]                #7.13
21          mov     DWORD PTR [-8+rbp], eax                 #7.11
22          mov     eax, 0                                  #8.12
23          leave                                           #8.12
24          ret                                             #8.12
```