|  |  |
| --- | --- |
|  | МИНОБРНАУКИ РОССИИ    федеральное государственное бюджетное образовательное учреждение  высшего образования  **«Балтийский государственный технический университет «ВОЕНМЕХ» им. Д.Ф. Устинова»**  **(БГТУ «ВОЕНМЕХ» им. Д.Ф. Устинова»)** |
| БГТУ.СМК-Ф-4.2-К5-01 |

Факультет И Информационные и управляющие системы

шифр наименование

Кафедра И5 Информационные системы и программные технологии

шифр наименование

Дисциплина Компьютерная практика

ЛАБОРАТОРНАЯ РАБОТА №2

на тему

Особенности использования отладчика GDB

Выполнил студент группы И582

Дубровский В. И.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | Фамилия И.О. | | | | |
|  |  |  |  |  | **РУКОВОДИТЕЛЬ** | | |
| Вальштейн |  |  | К.В. |  |  |  |  |
| Фамилия И.О. | |  | |  | Подпись | |  |
| Оценка | |  | |  |  |  |  |
| «\_\_\_\_\_» |  |  |  |  |  | 20\_\_\_\_ г. | |

САНКТ-ПЕТЕРБУРГ

2019 г.

**Цель работы**

Ознакомиться с особенностями использования отладчика GDB.

**Задание 1**

Найти число Фибоначчи, ближайшее к заданному натуральному числу N.

**Используемые команды:**

break 14 – установка точки остановки на строчке 14, continue **-** продолжить выполнение программы до следующей точки останова или ошибки, step - перейти к следующей строчке в программе.

**Текст программы:**

#include <iostream>

int main()

{

int n;

std::cout << "Input n: ";

std::cin >> n;

if(n < 1 || std::cin.fail()) return -1;

int fib1 = 1;

int fib2 = 2;

while(fib2 < n)

{

int temp = fib2;

fib2 = fib1 + fib2;

fib1 = temp;

}

int result = n-fib1 > fib2 - n ? fib2 : fib1;

std::cout << result << std::endl;

return 0;

}

**Результат отладки:**

На рисунках 1-3 мы видим значение вычисляемого выражения в цикле.

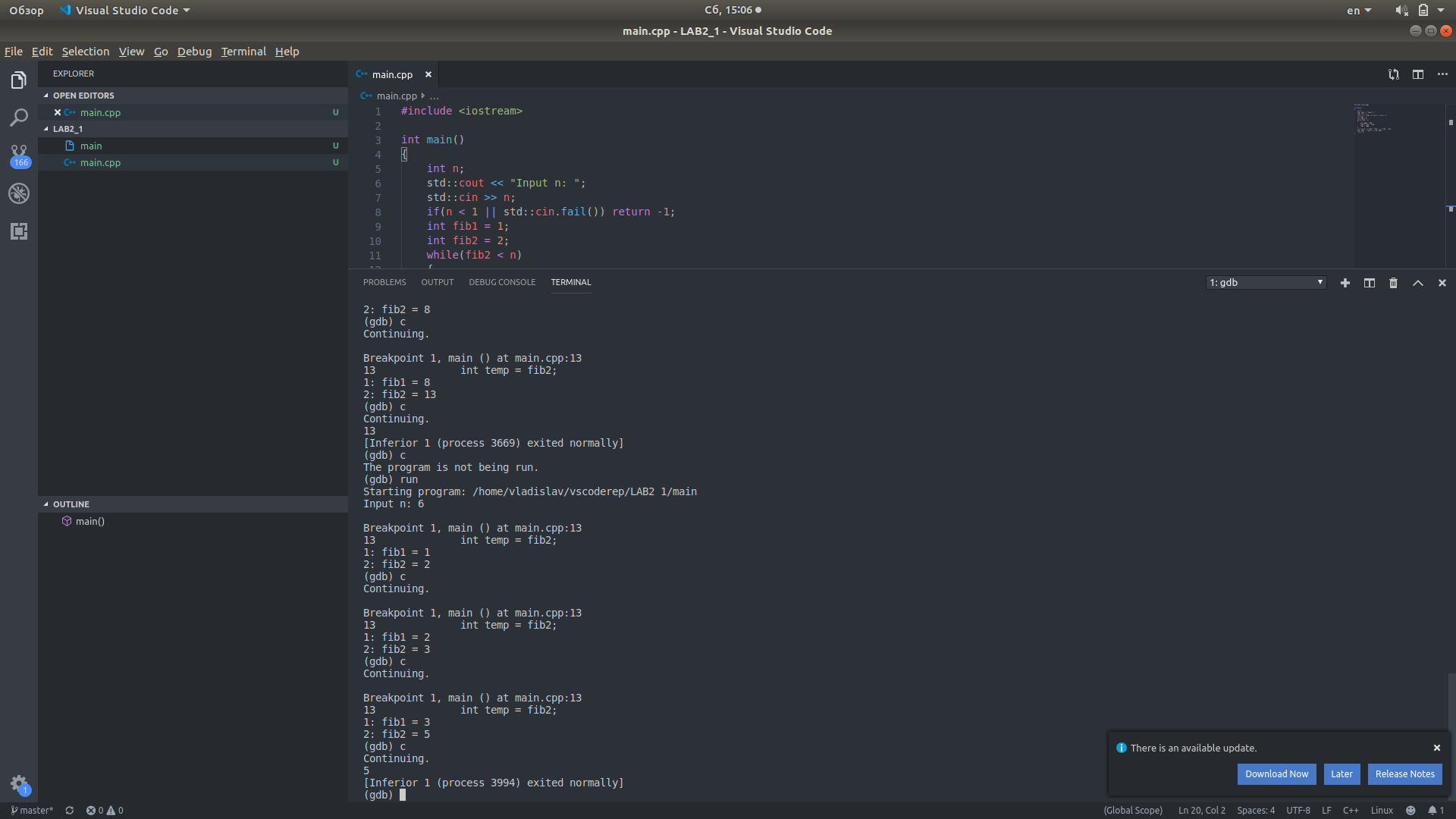


Рисунок 2.1 - Значение вычисляемого выражения на 1 шаге в теле цикла

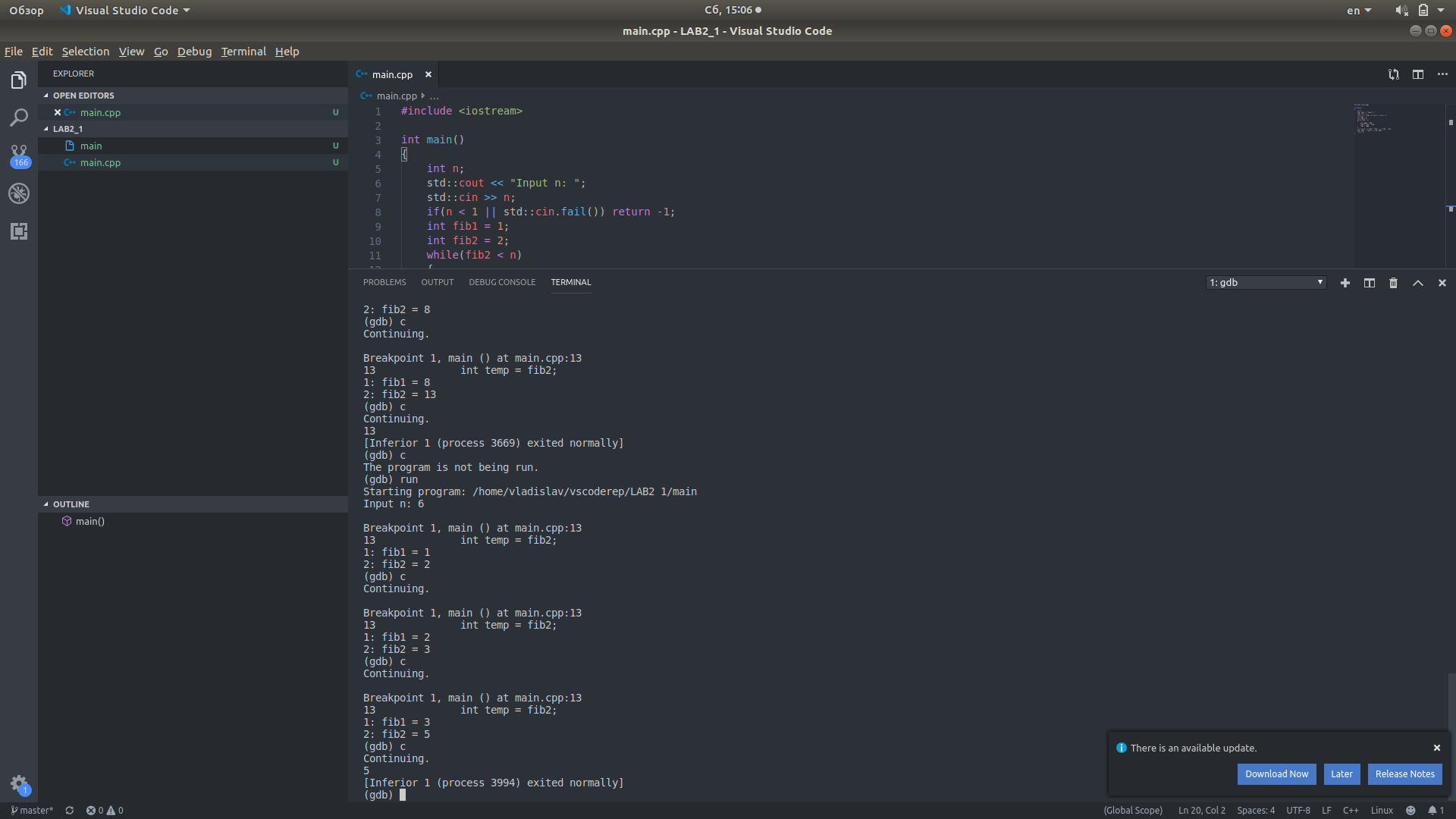


Рисунок 2.2 - Значение вычисляемого выражения на 2 шаге в теле цикла

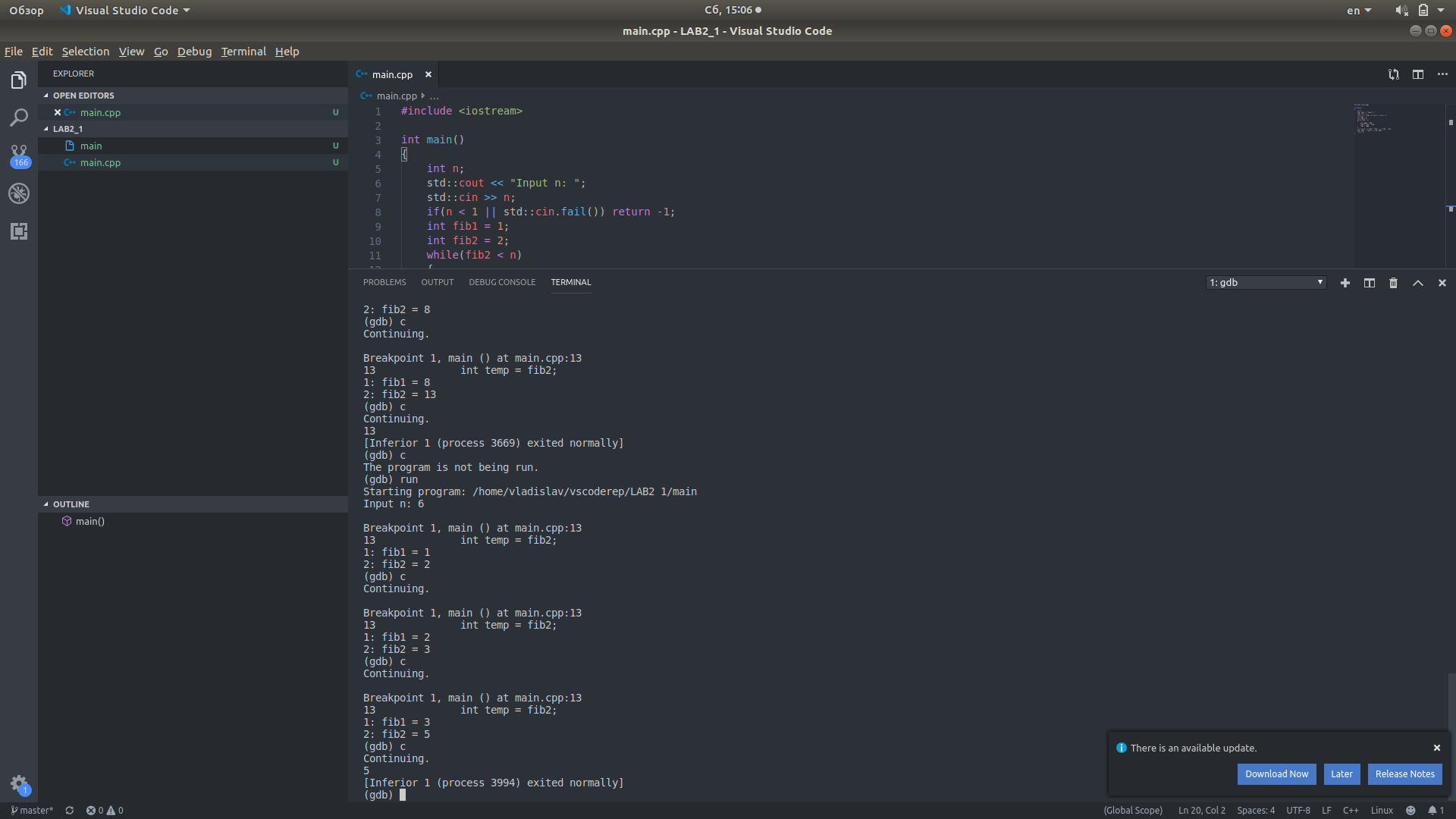


Рисунок 2.3 - Значение вычисляемого выражения на 3 шаге в теле цикла

**Задание 2**

Даны две строки. Составить третью строку из слов, имеющихся в обеих данных строках.

**Используемые команды:**

break inputStr, break outputStr, break 23, break 26, break 32 – установка точки остановки на строчках 5, 15, 23, 26, 32, where - для отображения содержимого стека, step **-** переход к следующей строчке программы.

**Результат отладки:**

На рисунке 8 представлен стек функций перед входом в функцию ввода, который содержит только главную функцию main.

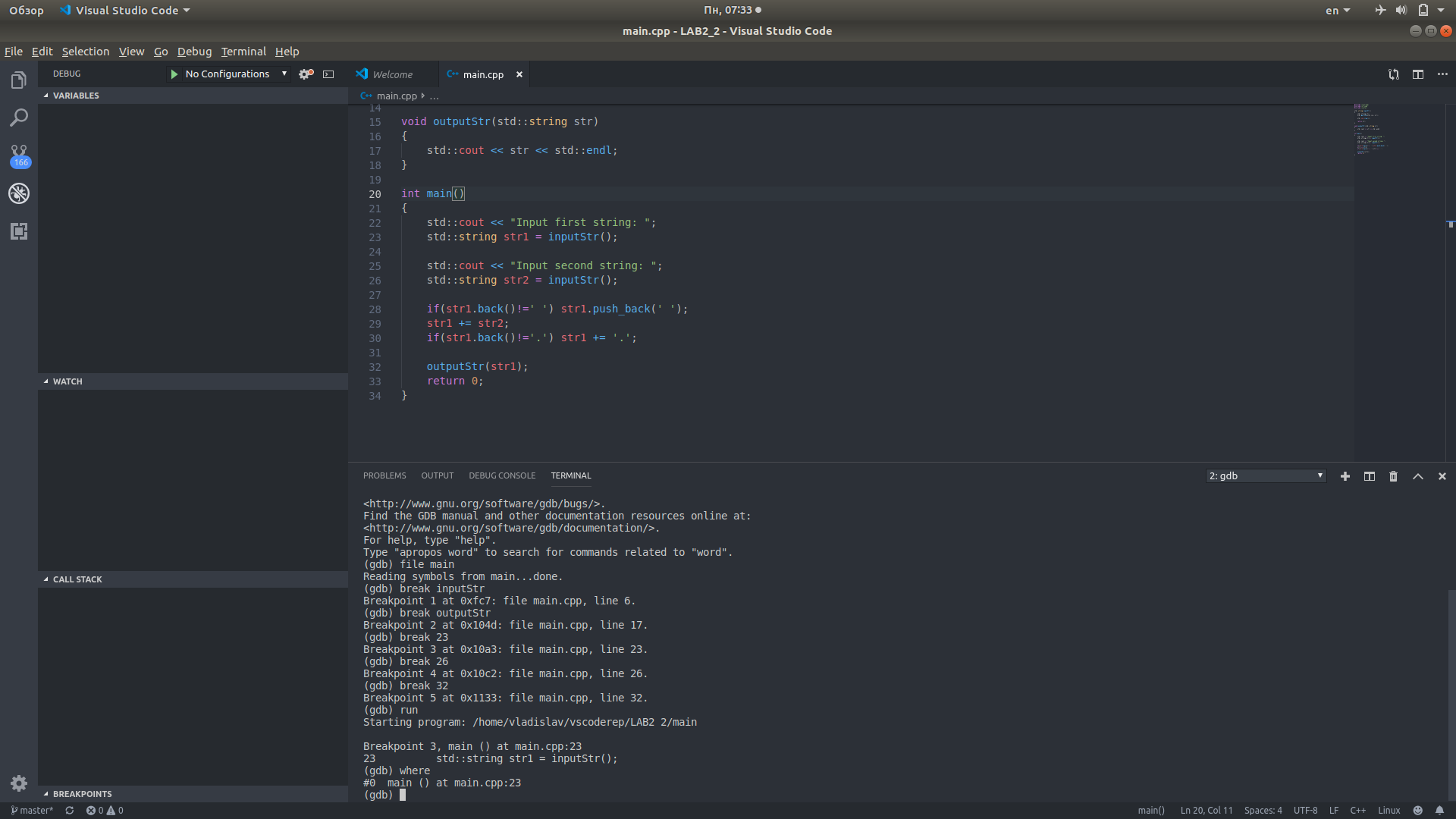


Рисунок 2.5 – Состояние стека на момент входа в функцию

На рисунке 2.6 мы видим стек функций после входа в функцию ввода, в стеке отображен адрес функции и входные данные.

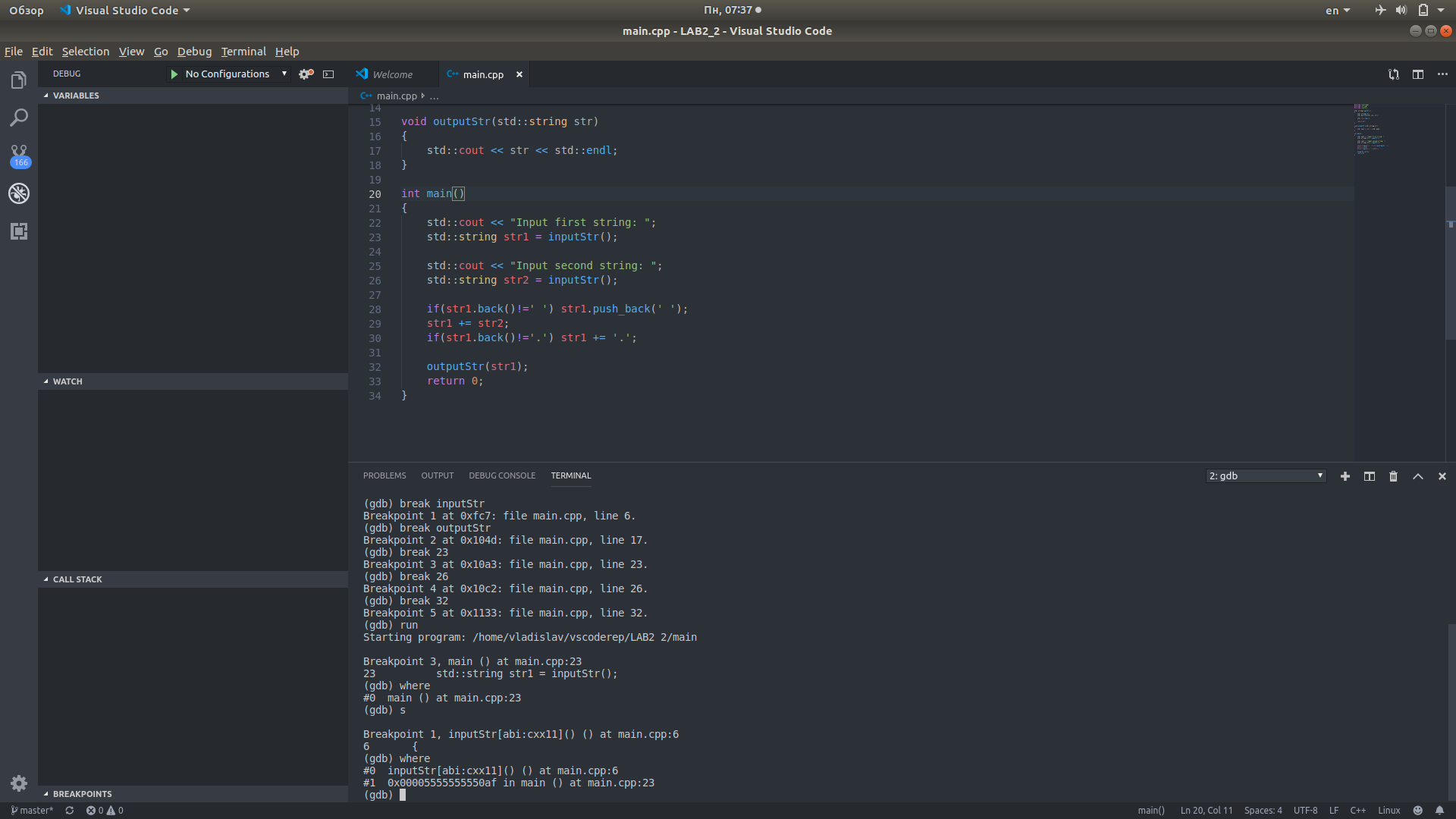


Рисунок 2.6 – Состояние стека после входа в функцию

На рисунке 2.7, 2.8 видно, что для функции вывода все аналогично.

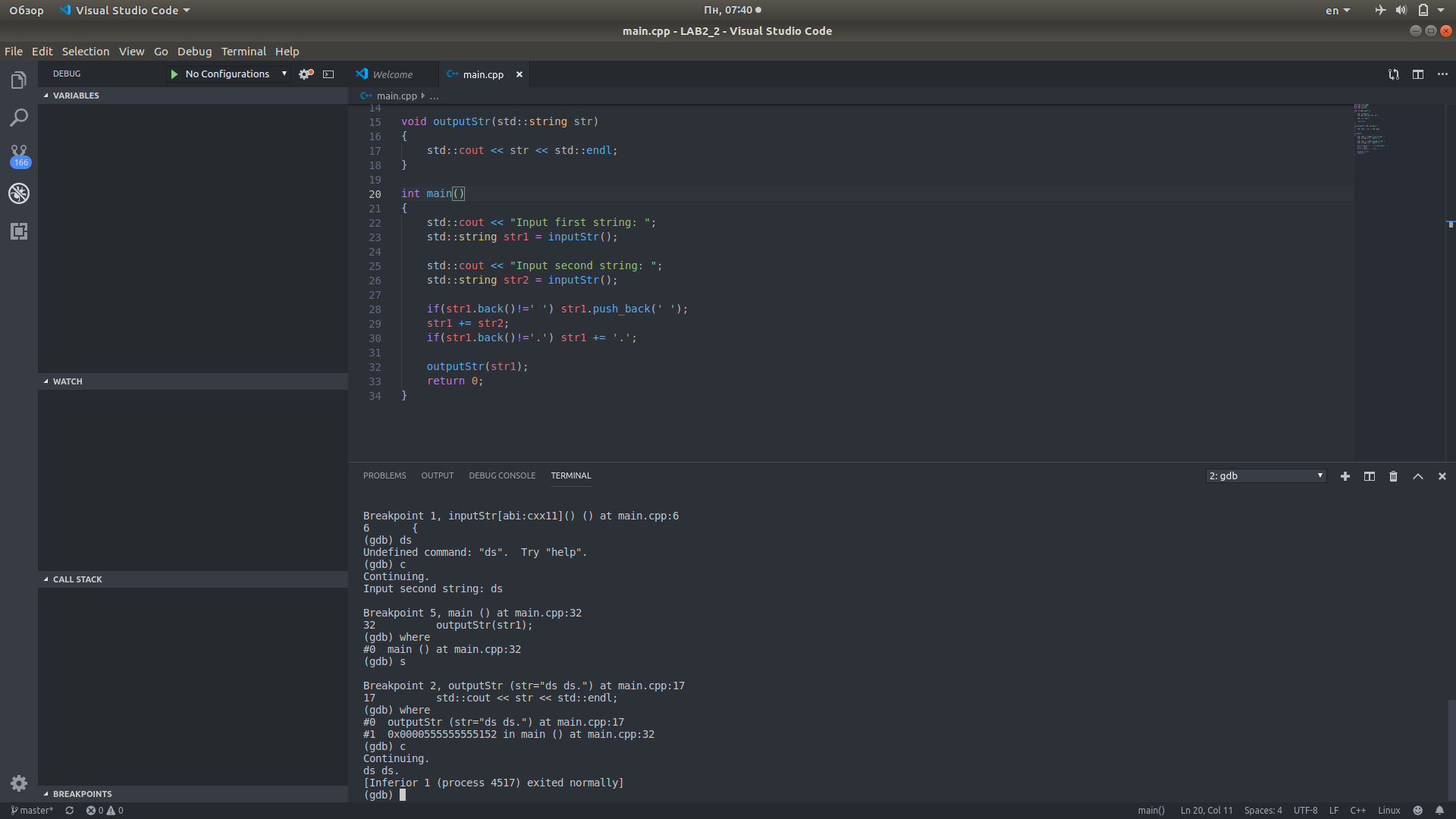


Рисунок 2.7 – Состояние стека на момент входа в функцию

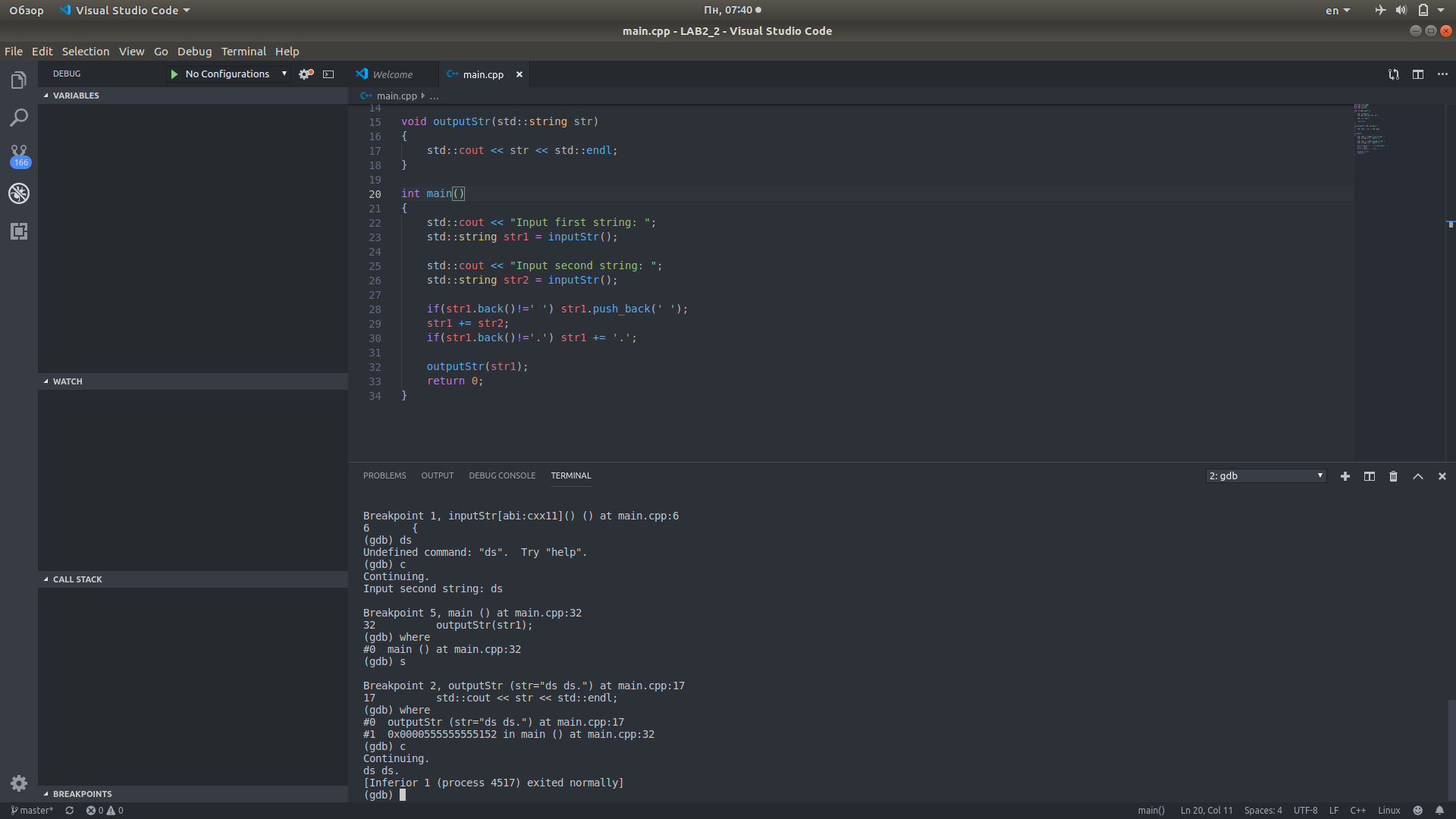


Рисунок 2.8 – Состояние стека после входа в функцию вывода.

**Текст программы:**

#include <iostream>

#include <string>

#include <vector>

std::string inputStr()

{

std::string str;

std::getline(std::cin, str);

std::cin.clear();

return str;

}

void outputStr(std::string str)

{

std::cout << str << std::endl;

}

int main()

{

std::cout << "Input first string: ";

std::string str1 = inputStr();

std::cout << "Input second string: ";

std::string str2 = inputStr();

if(str1.back()!=' ') str1.push\_back(' ');

str1 += str2;

if(str1.back()!='.') str1 += '.';

outputStr(str1);

return 0;

}

**Задание 3**

Дана строка символов. Выделить и вывести слова, ограниченные

пробелом или знаками препинания: запятая, точка, двоеточие,

точка с запятой.

**Используемые команды:**

break 7, break 26 – установка точки остановки на строчке 7 и 26, where - для отображения содержимого стека, step **-** продолжение для перехода к следующей строчке программы.

**Результат отладки**:

На рисунке 2.9 и 2.11 видно, что отладочная информация библиотеки ссылается на исходный файл.

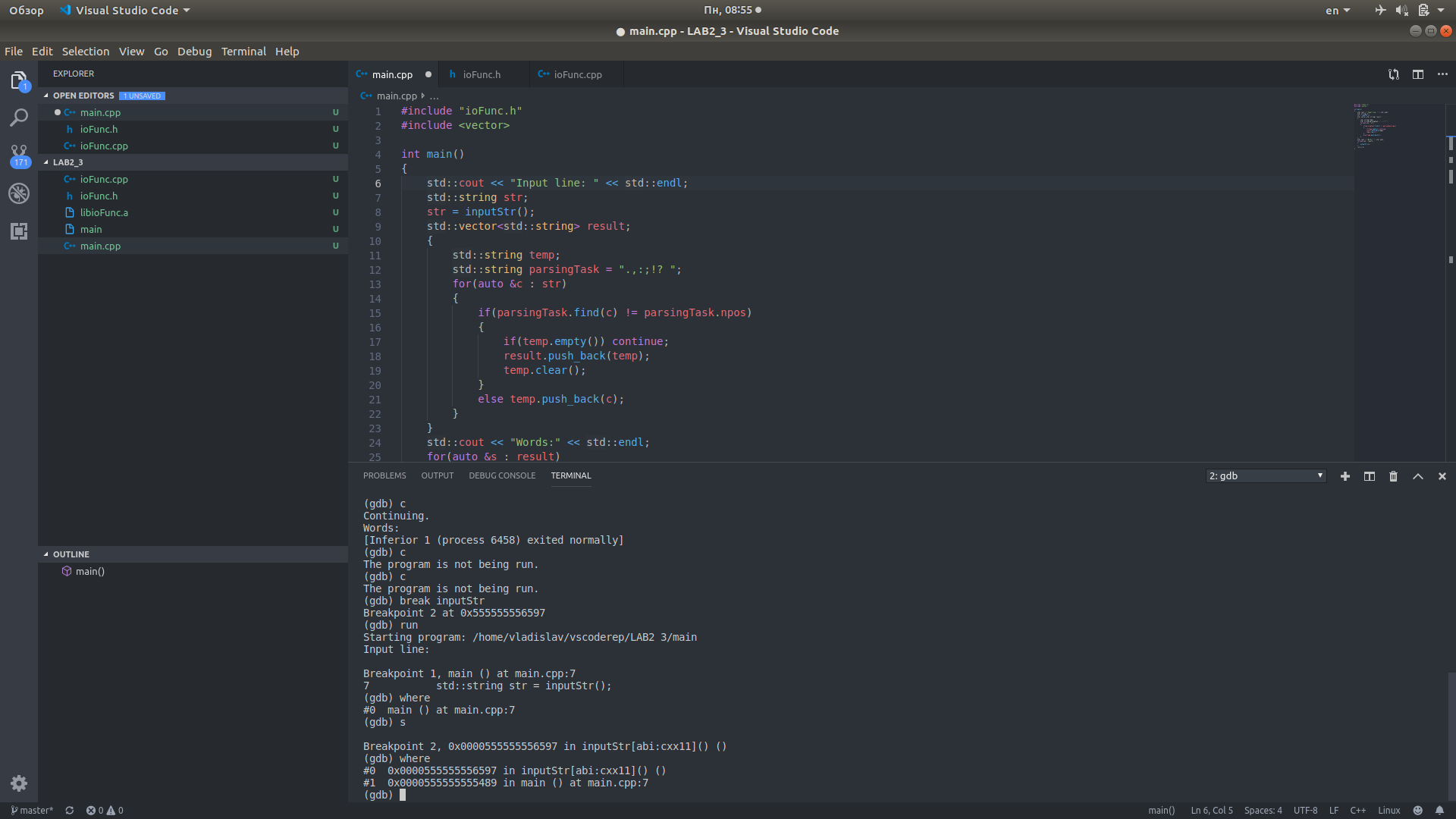


Рисунок 2.9 – Состояние стека до входа в функцию ввода библиотеки

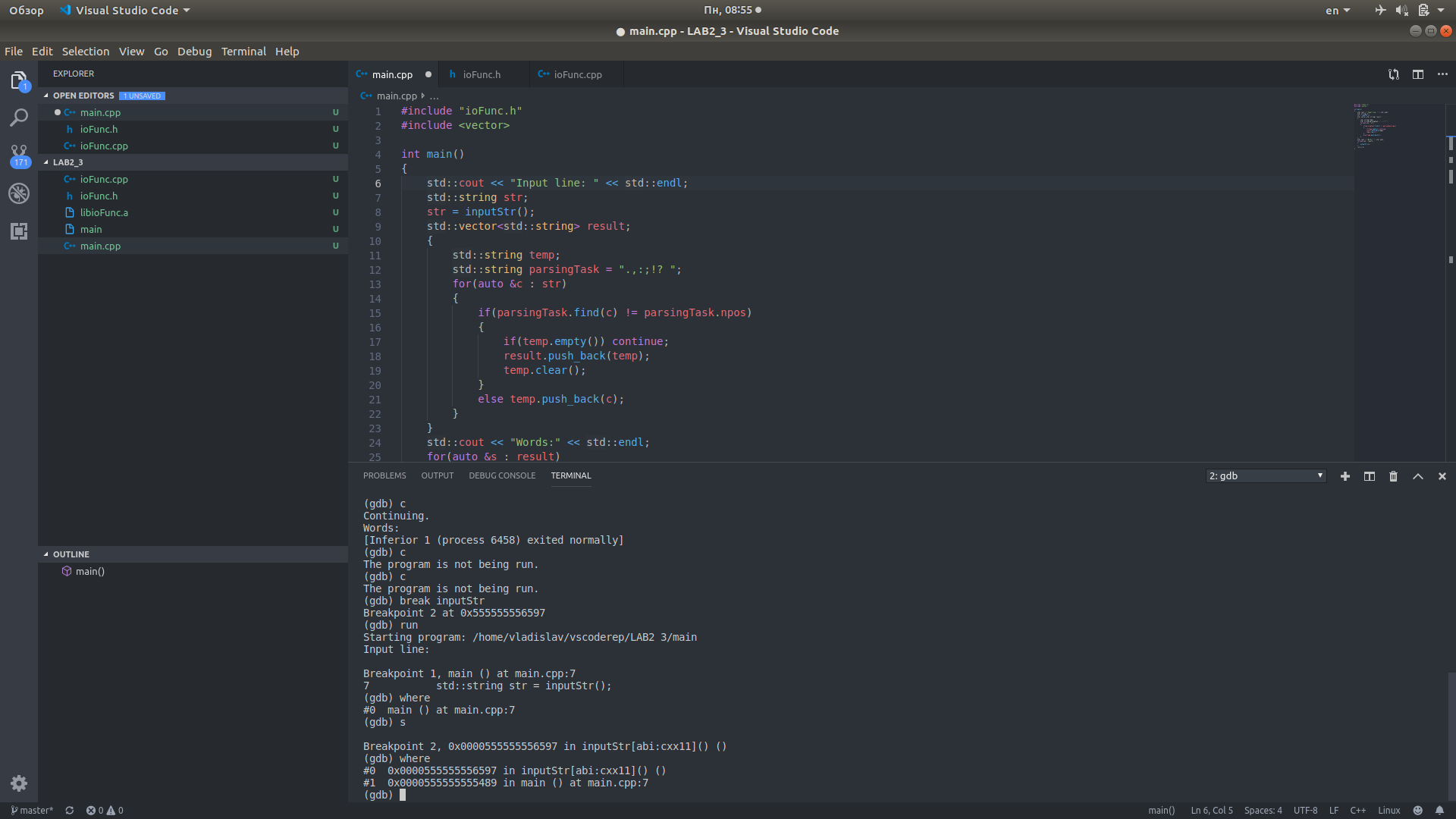


Рисунок 2.10 – Состояние стека после входа в функцию ввода библиотеки

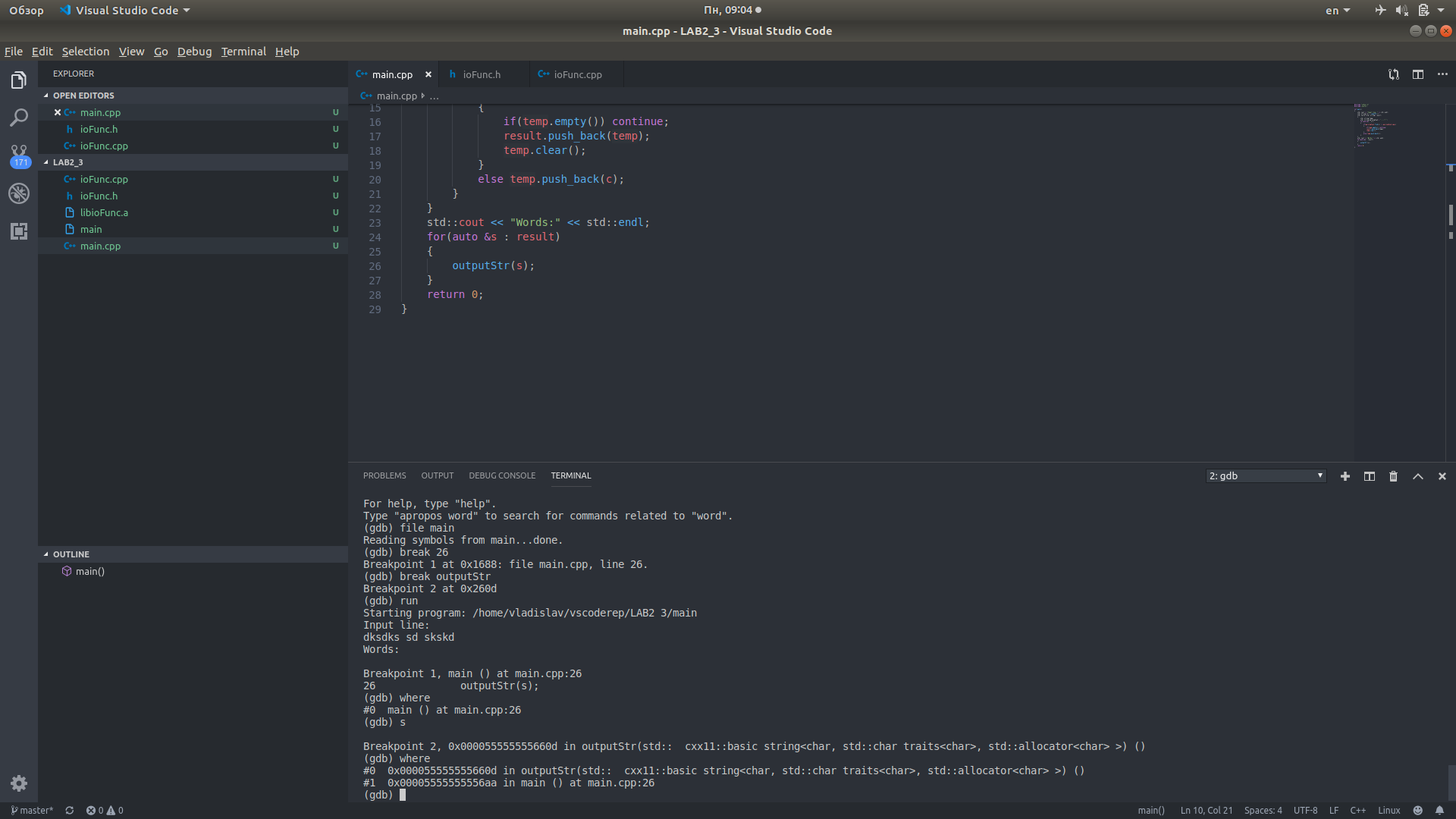


Рисунок 2.11 – Состояние стека до входа в функцию вывода библиотеки

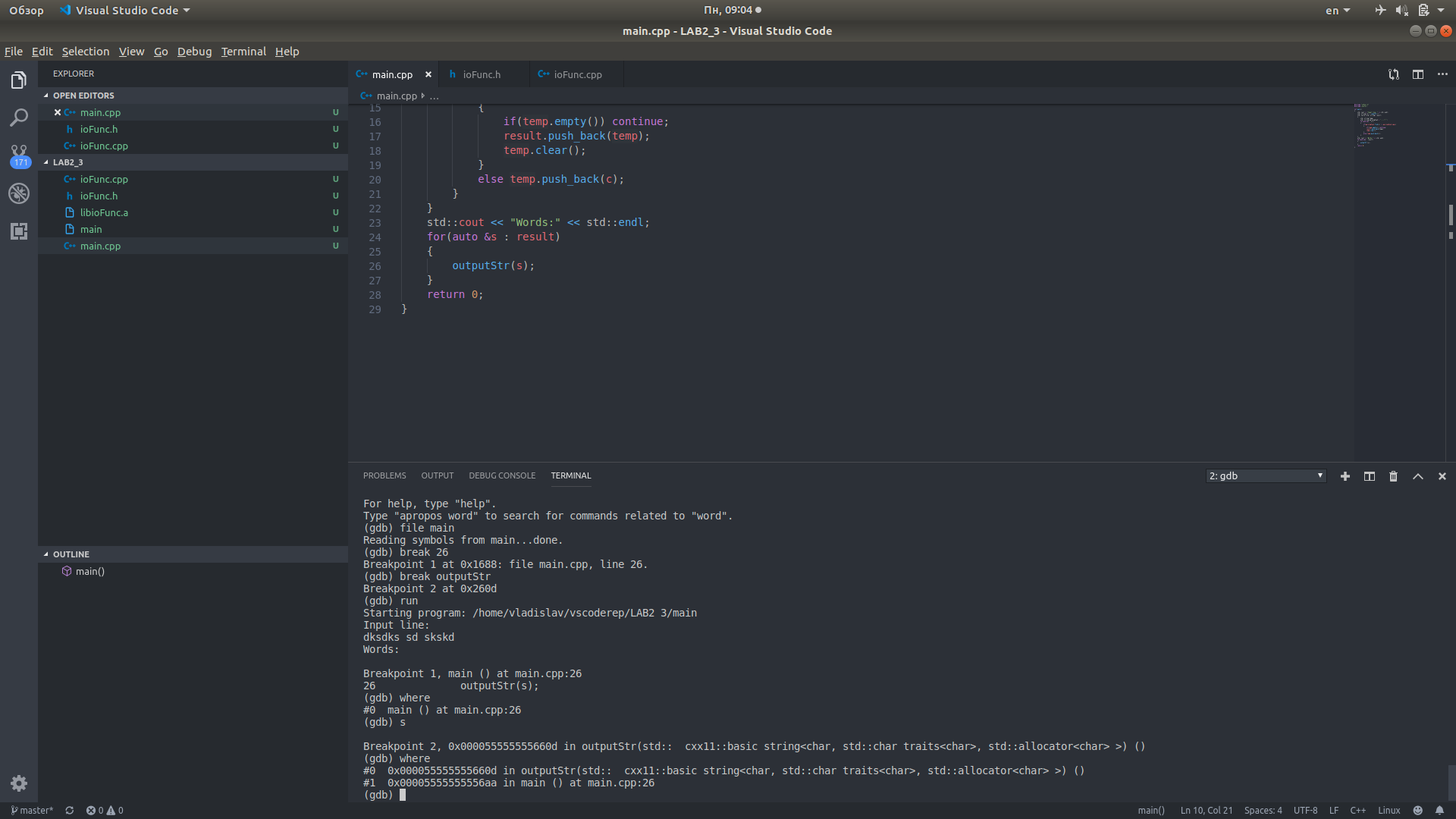


Рисунок 2.12 – Состояние стека после входа в функцию вывода библиотеки

Текст программы:

main.cpp

#include "ioFunc.h"

#include <vector>

int main()

{

std::cout << "Input line: " << std::endl;

std::string str = inputStr();

std::vector<std::string> result;

{

std::string temp;

std::string parsingTask = ".,:;!? ";

for(auto &c : str)

{

if(parsingTask.find(c) != parsingTask.npos)

{

if(temp.empty()) continue;

result.push\_back(temp);

temp.clear();

}

else temp.push\_back(c);

}

}

std::cout << "Words:" << std::endl;

for(auto &s : result)

{

outputStr(s);

}

return 0;

}

ioFunc.h

#pragma once

#include <iostream>

#include <string>

std::string inputStr();

void outputStr(std::string str);

ioFunc.cpp

#include "ioFunc.h"

std::string inputStr()

{

std::string temp;

std::getline(std::cin,temp);

return temp;

}

void outputStr(std::string str)

{

std::cout << str << std::endl;

}