МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ ИМЕНИ

ПАТРИСА ЛУМУМБЫ»

(РУДН)

Факультет Искусственного Интеллекта

факультет

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №**

Практикум по программированию

наименование дисциплины

Вариант №8

тема (вариант)

Преподаватель \_\_\_\_\_\_\_\_\_\_ В.С. Тынченко

подпись, дата инициалы, фамилия

Студент ЗФИмд01-24 1132249437 \_\_\_\_\_\_\_\_\_\_ А.В. Клюкин

номер группы номер студ. билета подпись, дата инициалы, фамилия

Москва 2025

**Цель работы**

приобретение навыков обработки произвольного текста.

**Задачи работы**

Из произвольного текста, содержащего не более 10 строк, в каждой строке не

более 80 символов, вывести на экран слова в порядке не возрастания их

длины.

**Основные этапы**

1. Изучить теоретические сведения к данной лабораторной работе (см.

прикрепленный файл " Теория Строки в Си")

1. Ознакомиться с постановкой задачи (см. прикрепленный

Файл "Варианты заданий 10", п. 10.1). Вариант задания соответствует вашему

номеру в списке группы.

1. Составить программу на языке С++ и выполнить ее отладку. Работать со

строкой как с символьным массивом. Тип string не использовать. См.

вложенный файл с примером программы «ПримерСтроки».

1. Отладить программу на подготовленных наборах тестовых данных.

**Листинг**

1. #include <algorithm>

2. #include <iostream>

3. #include <cstring>

4. #include <cstdlib>

5. #include <locale.h>

6. #include <conio.h>

7.

8. int main() {

9. const int M = 10;

10. char\* s[M], wstr[80];

11. char\*\* words = new char\* [40];

12. int kol\_str = 0;

13.

14. setlocale(0, "");

15. system("cls");

16.

17. // Ввод строк

18. while (kol\_str < M && gets\_s(wstr, 80) != NULL) {

19. s[kol\_str] = new char[strlen(wstr) + 1];

20. strcpy\_s(s[kol\_str], strlen(wstr) + 1, wstr);

21. kol\_str++;

22. }

23.

24. // Разделение на слова

25. int e = 0;

26. bool inword = false;

27. for (int i = 0; i < kol\_str; i++) {

28. int len = strlen(s[i]);

29. int word\_start = 0;

30.

31. for (int j = 0; j <= len; j++) {

32. if (s[i][j] == ' ' || s[i][j] == ',' || s[i][j] == '\0') {

33. if (inword) {

34. int word\_len = j - word\_start;

35. words[e] = new char[word\_len + 1];

36. strncpy\_s(words[e], word\_len + 1, s[i] + word\_start, word\_len);

37. words[e][word\_len] = '\0'; // Обязательно добавляем терминатор!

38. e++;

39. inword = false;

40. }

41. }

42. else {

43. if (!inword) {

44. word\_start = j;

45. inword = true;

46. }

47. }

48. }

49. }

50.

51. //вывод слов по длинне

52.

53. //std::sort(words, words + e, [](const char\* a, const char\* b) {

54. // return std::strlen(a) < std::strlen(b);

55. // });

56.

57. //for (int i = 0; i < e; i++) {

58. // std::cout << words[i] << "\n";

59. //}

60.

61.

62. for (int l = 1; l <= 80; ++l) { // Начинаем с длины 1

63. for (int i = 0; i < e; ++i) {

64. if (std::strlen(words[i]) == l) {

65. std::cout << words[i] << '\n';

66. }

67. }

68. }

69.

70.

71. // Освобождение памяти

72. for (int i = 0; i < kol\_str; i++) {

73. delete[] s[i];

74. }

75.

76. for (int i = 0; i < e; i++) {

77. delete[] words[i];

78. }

79. delete[] words;

80.

81. return 0;

82. }

83.