*Федеральное государственное бюджетное образовательное учреждение* *высшего образования*

|  |  |
| --- | --- |
|  | ***«Московский государственный технический университет  имени Н.Э. Баумана»***  ***(национальный исследовательский университет)***  ***(МГТУ им. Н.Э. Баумана)*** |

ФАКУЛЬТЕТ \_\_\_\_ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ\_\_\_\_\_\_\_\_

КАФЕДРА \_\_\_\_\_\_Компьютерные Системы и сети (ИУ6)\_\_\_\_\_\_\_\_\_\_\_

**Отчет**

**по лабораторной работе № 6**

**Название лабораторной работы: Динамические структуры данных. Деревья**

**Дисциплина: Объектно-ориентированное программирование**

Студент гр. ИУ6-22Б  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. П. Плютто**



(Подпись, дата) (И.О. Фамилия)



Преподаватель  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ О. А. Веселовская**

(Подпись, дата) (И.О. Фамилия)

Москва, 2022

Задание

Реализовать программу сортировки чисел с использованием дерева. Обеспечить вывод элементов по возрастанию и убыванию.

1. Создадим структуру, для построения бинарного дерева

1. struct binTree

2. {

3. binTree \*left;

4. binTree \*right;

5. int data;

6. };

1. Объявим и опишем функцию добавления элемента в бинарное дерево

1. binTree \*insert(int item, binTree \*root)

2. {

3. binTree \*p;

4. p = new binTree;

5. binTree \*parent;

6. p->data = item;

7. p->left = NULL;

8. p->right = NULL;

9. parent = NULL;

10. if (root->data == -1)

11. {

12. root = p;

13. }

14. else

15. {

16. binTree \*ptr;

17. ptr = root;

18. while (ptr != NULL)

19. {

20. parent = ptr;

21. if (item > ptr->data)

22. ptr = ptr->right;

23. else

24. ptr = ptr->left;

25. }

26. if (item < parent->data)

27. parent->left = p;

28. else

29. parent->right = p;

30. }

31. return root;

32. }

1. Объявим и опишем функцию печати бинарного дерева (печать производится рекурсивно)

1. void printBinTree(binTree \*ptr, bool reverse)

2. {

3. if (ptr != NULL)

4. {

5. if (reverse)

6. {

7. printBinTree(ptr->right, reverse);

8. cout << ptr->data << " ";

9. printBinTree(ptr->left, reverse);

10. }

11. else

12. {

13. printBinTree(ptr->left, reverse);

14. cout << ptr->data << " ";

15. printBinTree(ptr->right, reverse);

16. };

17. }

18. }

1. Создадим тестирующую функцию с возможностью ввода значений с клавиатуры.

1. int main()

2. {

3. binTree \*b;

4. b = new binTree;

5. b->data = -1;

6. int tmp{};

7. while (tmp != -1)

8. {

9. cin >> tmp;

10. if (tmp != -1)

11. b = insert(tmp, b);

12. };

13. printBinTree(b, 0);

14. cout << "\n";

15. printBinTree(b, 1);

16. }

17.

Полный код программы и ее вывод с тестирующими данными приведен ниже.

1. #include "iostream"

2. using std::cin, std::cout;

3.

4. struct binTree

5. {

6. binTree \*left;

7. binTree \*right;

8. int data;

9. };

10.

11. binTree \*insert(int item, binTree \*root)

12. {

13. binTree \*p;

14. p = new binTree;

15. binTree \*parent;

16. p->data = item;

17. p->left = NULL;

18. p->right = NULL;

19. parent = NULL;

20. if (root->data == -1)

21. {

22. root = p;

23. }

24. else

25. {

26. binTree \*ptr;

27. ptr = root;

28. while (ptr != NULL)

29. {

30. parent = ptr;

31. if (item > ptr->data)

32. ptr = ptr->right;

33. else

34. ptr = ptr->left;

35. }

36. if (item < parent->data)

37. parent->left = p;

38. else

39. parent->right = p;

40. }

41. return root;

42. }

43.

44. void printBinTree(binTree \*ptr, bool reverse)

45. {

46. if (ptr != NULL)

47. {

48. printBinTree(reverse ? ptr->right : ptr->left, reverse);

49. cout << ptr->data << " ";

50. printBinTree(reverse ? ptr->left : ptr->right, reverse);

51. }

52. }

53.

54. int main()

55. {

56. binTree \*b;

57. b = new binTree;

58. b->data = -1;

59. int tmp{};

60. while (tmp != -1)

61. {

62. cin >> tmp;

63. if (tmp != -1)

64. b = insert(tmp, b);

65. };

66. printBinTree(b, 0);

67. cout << "\n";

68. printBinTree(b, 1);

69. }

70.

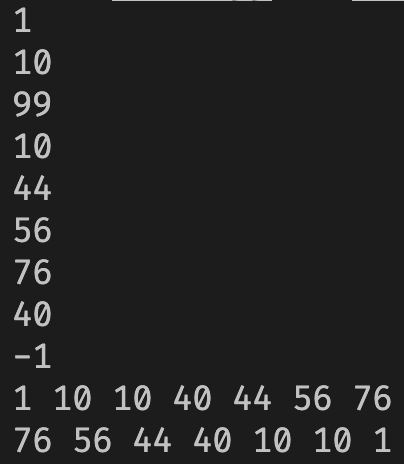


Рисунок : Вывод программы



Рисунок : Схемы алгоритмов для тестирующей программы, функции добавления и печати

Вывод: *В ходе выполнения работы были изучены и применены на практике динамические структуры данных (деревья) в языке C++.*

*Была разработана структура для хранения узла дерева, а также функции и процедуры для работы с деревом.*