|  |  |
| --- | --- |
| Министерство образования Республики Беларусь  Учреждение образования «Полоцкий государственный университет» | |
|  | Факультет информационных технологий  Кафедра технологий программирования |
| Лабораторная работа №1  По дисциплине «О-ОТПиСП»  «Классы и объекты в c++» | |
| Выполнил: | Кригина Е. А.  Группа 19-ИТ-1 |
| Проверил: | Васильева. Д. М. |
| Полоцк  2021г. | |

**Цель:** получить практические навыки реализации классов на С++

**Задание:** реализовать пользовательский класс «животное» содержащий в себе поля: char\* name - название, char\* clas - класс, int aWeight – средний вес. Также требуется определить в классе следующее: конструкторы (без параметров, с параметрами, копирования); деструктор; компоненты-функции для просмотра и установки полей данных, указатель на компоненту-функцию, указатель на экземпляр объекта.

Определение пользовательского класса с конструкторами и деструктором. А так же задейсвуем две переменные в статической и динамической памяти. В статической памяти будет находится свойство *Name[25]* так как это массив с определенным значением. В динамической памяти будет располагаться динамический массив *clas* так как это указатель. Так же в методе установки значения будем изменять память при помощи функции *realloc* динамическому массиву *clas*.

Реализация данного класса:

1. class animal { // инициализировали класс с именем animal
2. public: // инициализируем поля, конструкторы, деструктор, функции класса с доступом public
3. char name[25]; //название животного
4. char \*clas; //класс животного
5. int aWeight; //средний вес
6. animal() { // определение конструктора без параметров (ничего не принимает)
7. strcpy(name, "unknown\t"); // полю name присваиваем “unknown” в статической памяти
8. class = NULL// присваиваем полю clas NULL в динамической памяти
9. aWeight = 0; // присваиваем полю aWeight 0
10. cout << "\nВызвался конструктор без параметров!\n";
11. };
12. animal(const char \*\_name, const char \* \_clas, int \_weight) { // определили конструктор с параметрами
13. strcpy(name, \_name); // присваиваем полю name класса принимаемое конструктором значение
14. clas = (char\*)malloc(sizeof(\_clas)); //дополнительно выделим память свойсву класса размером с передаваемый объект
15. strcpy(clas, \_clas); // присваеваем
16. aWeight = \_weight; //устанавливаем вес
17. cout << "\nВызвался конструтор с парметрами\n";
18. };
19. animal(animal &al) { //определили конструктор копирования (принимает экземпляр того же класса)
20. strcpy(name, al.name); // копируем значение полей принимаемого экземпляра
21. clas = (char\*)malloc(sizeof(al.clas)); //дополнительно выделим память свойсву класса размером с элемент передаваемого объекта
22. strcpy(clas, al.clas);
23. aWeight = al.aWeight;
24. cout << "\nВызвался конструктор копирования\n";
25. };
26. ~animal() { // определили деструктор
27. cout << "\nВызвался деструктор\n";
28. }
29. char\* GetName() { return name; } // определили функцию просмотра значения поля (геттеры)
30. char\* GetClas() { return clas; }
31. int GetAWeight() { return aWeight; }
32. //функции установки значения (сеттеры)
33. void SetName(const char\* \_name) {
34. strcpy(name, \_name);
35. }
36. void SetClas(const char\* \_clas) {
37. clas = (char\*)realloc(clas, sizeof(\_clas)); // изменим размер памяти под передаваемый элемент
38. strcpy(clas, \_clas); //присвоим новое значение
39. }
40. void SetAWeight(int weight) {
41. aWeight = weight;
42. }void PrintAnimal() { // функция вывода экземпляра класса
43. char result[100];
44. sprintf(result, "\n\nИнформация о животном\nИмя: %s \nКласс: %s \nСредний вес: %d\n\n", name, clas, aWeight);
45. cout << result;
46. }
47. };

Определение указателя на компоненту-функцию:

1. void example() { // определили функцию, на которую будет ссылаться указатель fncPtr
2. cout << "\n\nУказатель fcnPtr ссылается на функцию example\n";
3. }
4. void (\*fcnPtr)() = example; // Указатель fcnPtr не принимает аргументов и ничего не возвращает, указывает на функцию example

Определение указателя на экземпляр класса:

animal\* cat3 = &cat; // определили указатель cat3, который ссылается на участок памяти объекта cat

main():

1. int main() {
2. setlocale(LC\_ALL, "");
3. animal cat; // вызвали конструктор без параметров (ничего не передали)
4. cat.PrintAnimal();
5. animal cat1("Kod", "Mlekopit", 6); // вызвали конструктор с параметрами
6. cat1.PrintAnimal();
7. animal cat2(cat1); // вызвали конструктор копирования (копируем объект cat1 в объект cat2)
8. cat2.PrintAnimal();
9. (\*fcnPtr)(); // вызвали функцию example с помощью указателя fcnPtr, ссылающегося на эту функцию
10. animal\* cat3 = &cat;
11. cat3->PrintAnimal(); // разыменовываем из указателя объект cat3 и выводим его
12. }

Ход выполнения данной лабораторной работы описан в комментариях листингов.

Результат выполнения лабораторной работы:

