Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЁТ**

**по лабораторной работе №18.2**

Дисциплина: основы алгоритмизации и программирования

Тема: “ Классы и объекты. Использование конструкторов”

Вариант 4

Выполнил:

Студент группы ИВТ-20-2б

Брейкин Алексей Дмитриевич

Проверил:

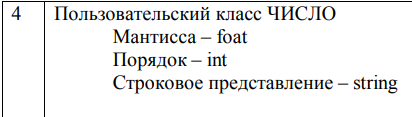
Доцент кафедры ИТАС

Полякова О. А.

**Пермь, 2020**

**Постановка задачи**

1. Определить пользовательский класс.
2. Определить в класс следующие конструкторы: без параметров, с параметрами, копирования.
3. Определить в классе деструктор.

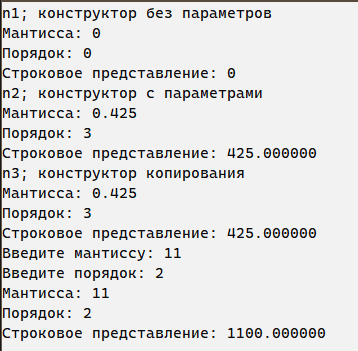
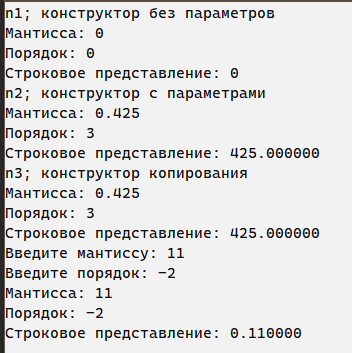


**Исходный код**

1. #include <iostream>
2. #include "number.h"
4. using namespace std;
6. void Show(Number num) {
7. num.Show();
8. }
10. int main() {
11. system("chcp 1251>nul");
12. cout << "n1; конструктор без параметров" << endl;
13. Number n1;
14. n1.Show();
16. cout << "n2; конструктор с параметрами" << endl;
17. Number n2(0.425, 3);
18. n2.Show();
20. cout << "n3; конструктор копирования" << endl;
21. Number n3 = n2;
22. n3.Show();
23. double mantissa;
24. int poryadok;
25. cout << "Введите мантиссу: "; cin >> mantissa;
26. cout << "Введите порядок: "; cin >> poryadok;
27. n3.setMantissa(mantissa);
28. n3.setPoryadok(poryadok);
29. Show(n3);
30. return 0;
31. }
32. #include "number.h"
33. #include <iostream>
34. #include <math.h>
36. using namespace std;
38. Number::Number() {
39. m\_mantissa = 0.0;
40. m\_poryadok = 0;
41. m\_number = "0";
42. }
44. Number::Number(double mantissa, int poryadok) {
45. m\_mantissa = mantissa;
46. m\_poryadok = poryadok;
47. m\_number = to\_string(m\_mantissa \* pow(10, m\_poryadok));
48. }
50. Number::Number(const Number& n) {
51. m\_mantissa = n.m\_mantissa;
52. m\_poryadok = n.m\_poryadok;
53. m\_number = n.m\_number;
54. }
56. void Number::updateNumber() {
57. m\_number = to\_string(m\_mantissa \* pow(10, m\_poryadok));
58. }
60. double Number::getMantissa() {
61. return m\_mantissa;
62. }
64. int Number::getPoryadok() {
65. return m\_poryadok;
66. }
68. string Number::getNumber() {
69. return m\_number;
70. }
72. void Number::setMantissa(double man) {
73. m\_mantissa = man;
74. updateNumber();
75. }
77. void Number::setPoryadok(int por) {
78. m\_poryadok = por;
79. updateNumber();
80. }

83. void Number::Show() {
84. cout << "Мантисса: " << m\_mantissa << endl;
85. cout << "Порядок: " << m\_poryadok << endl;
86. cout << "Строковое представление: " << m\_number << endl;
87. }
88. #include <string>
89. #pragma once
90. using namespace std;
92. class Number {
93. double m\_mantissa;
94. int m\_poryadok;
95. string m\_number;
97. void updateNumber();
98. public:
99. // конструкторы
100. Number();
101. Number(double, int);
102. Number(const Number&);
104. // геттеры
105. double getMantissa();
106. int getPoryadok();
107. string getNumber();
109. // сеттеры
110. void setMantissa(double);
111. void setPoryadok(int);
113. void Show();
114. };

**Скриншоты работы программы**

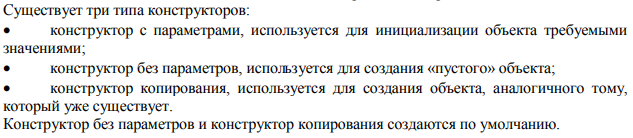
**** ****

**Контрольные вопросы**



1. Конструктор необходим для инициализации объекта.



2. 

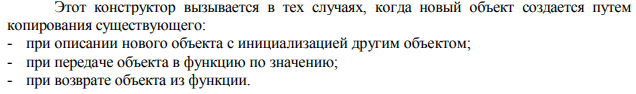


3. Деструктор – это особый вид метода, применяющийся для освобождения ресурсов, выделенных конструктором объекту. Описывать деструктор явным образом требуется в случае, когда объект содержит указатели на память, выделяемую динамически.



4. Конструктор с параметрами используется для инициализации объекта требуемыми значениями. Конструктор без параметров используется для создания «пустого» объекта. Конструктор копирования – это специальный вид конструктора, получающий в качестве единственного параметра указатель на объект этого же класса.

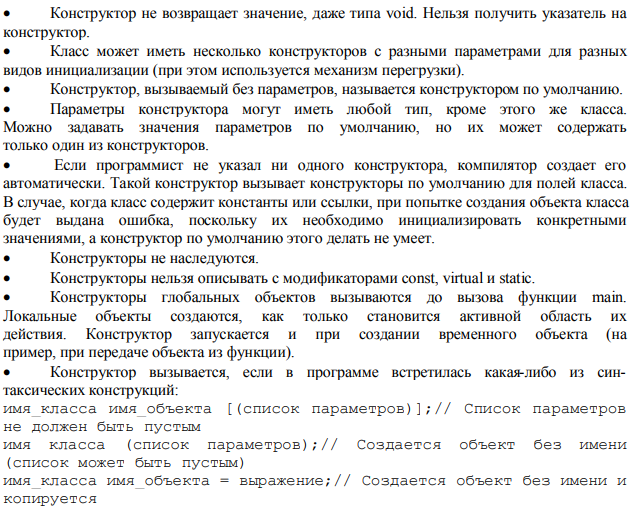


5. 

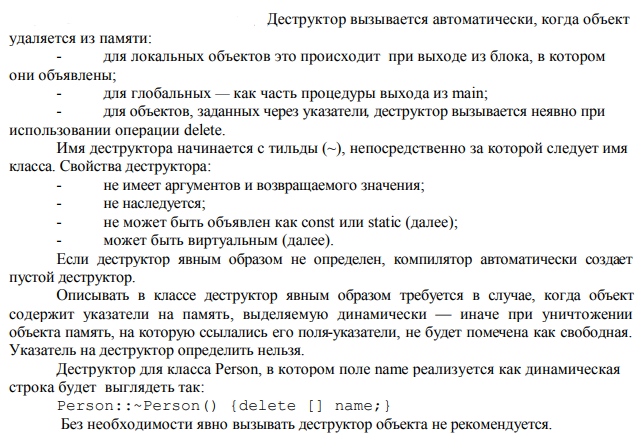


15

6.





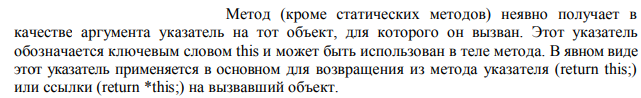
7. 

16

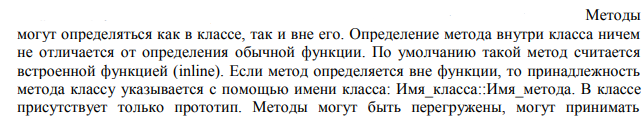


8.



9.



10.



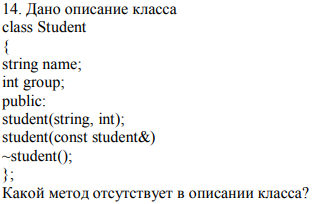
11. Конструктор не возвращает значения



12. Конструкторы и деструкторы без параметров, если таковые не были прописаны явно



13. Деструктор не возвращает значения



14. Отсутствует конструктор без параметров в описании класса

17



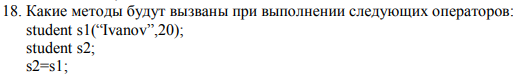
15. Будет вызван конструктор без параметров



16. Будет вызван конструктор с параметрами



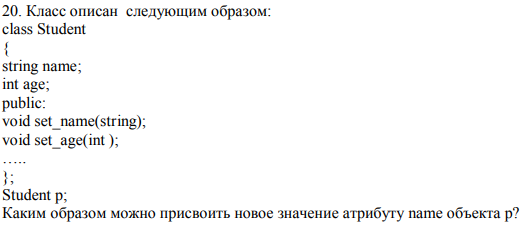
17. Будут вызваны конструктор с параметрами, а затем конструктор копирования



18. Будут вызваны конструктор с параметрами, затем конструктор без параметров, а затем конструктор копирования



19. Конструктор без параметров



20. C помощью метода set\_name()