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 №18.3**

Дисциплина: основы алгоритмизации и программирования

Тема: “ Классы и объекты. Перегрузка операций”

Вариант 4

Выполнил:

Студент группы ИВТ-20-2б

Брейкин Алексей Дмитриевич

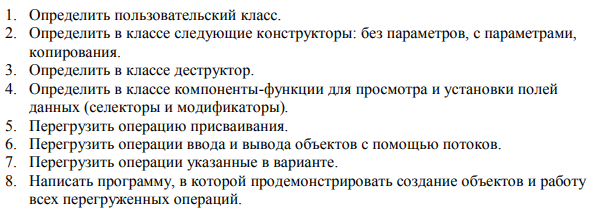
Проверил:

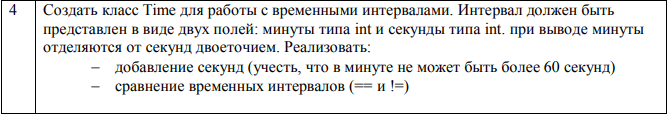
Доцент кафедры ИТАС

Полякова О. А.

**Пермь, 2020**

**Постановка задачи**

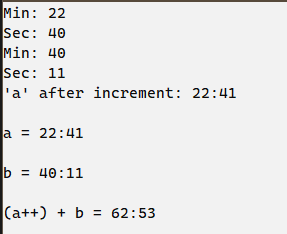




**Исходный код**

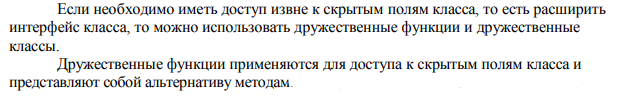
1. #include <iostream>
2. #include "Time.h"
4. using namespace std;
6. int main() {
7. system("chcp 1251>nul");
8. Time a;
9. Time b;
10. Time c;
11. cin >> a;
12. cin >> b;
13. ++a;
14. cout << "'a' after increment: ";
15. cout << a << endl;
16. c = (a++) + b;
17. cout << "a = "<< a << endl;
18. cout << "b = "<< b << endl;
19. cout << "c = "<< c << endl;
20. return 0;
21. }
23. #pragma once
24. #include <iostream>
25. using namespace std;
27. class Time {
28. int m\_mins;
29. int m\_secs;
31. void correctOverflow() {
32. int mins = 0;
33. while (m\_secs >= 60) {
34. m\_secs -= 60;
35. mins += 1;
36. }
37. m\_mins += mins;
38. }
39. public:
41. void addSeconds(int seconds) {
42. m\_secs += seconds;
43. correctOverflow();
44. }
46. Time& operator =(const Time&);
47. Time& operator ++();
48. Time operator ++(int);
50. Time operator +(const Time&);
52. friend ostream& operator <<(ostream&, Time&);
53. friend istream& operator >>(istream&, Time&);
54. };
56. #include "Time.h"
58. using namespace std;
60. istream& operator >>(istream& in, Time& p)
61. {
62. p.m\_mins = -1;
63. p.m\_secs = -1;
64. while (p.m\_mins < 0) {
65. cout << "Min: ";
66. cin >> p.m\_mins;
67. }
68. while (p.m\_secs < 0) {
69. cout << "Sec: ";
70. cin >> p.m\_secs;
71. }
72. p.correctOverflow();
73. return in;
74. }
76. ostream& operator <<(ostream& out, Time& p)
77. {
78. if (p.m\_mins < 10) cout << 0;
79. cout << p.m\_mins << ":";
80. if (p.m\_secs < 10) cout << 0;
81. cout << p.m\_secs << endl;
82. return out;
83. }
85. Time& Time::operator++()
86. {
87. this->m\_secs++;
88. this->correctOverflow();
89. return \*this;
90. }
92. Time Time::operator++(int)
93. {
94. Time temp = \*this;
95. ++temp.m\_secs;
96. temp.correctOverflow();
97. return temp;
98. }
100. Time Time::operator+(const Time& t)
101. {
102. Time n;
103. n.m\_mins = m\_mins + t.m\_mins;
104. n.m\_secs = m\_secs + t.m\_secs;
105. n.correctOverflow();
106. return n;
107. }
109. Time& Time::operator=(const Time& t)
110. {
111. if (&t == this) return \*this;
112. m\_mins = t.m\_mins;
113. m\_secs = t.m\_secs;
114. return \*this;
115. }

**Скриншоты работы программы**

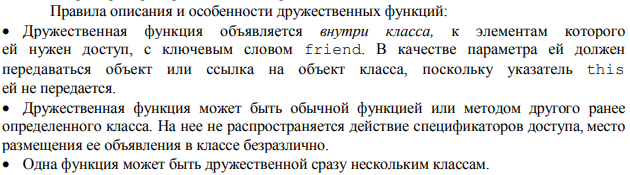
****

**Контрольные вопросы**



1. 

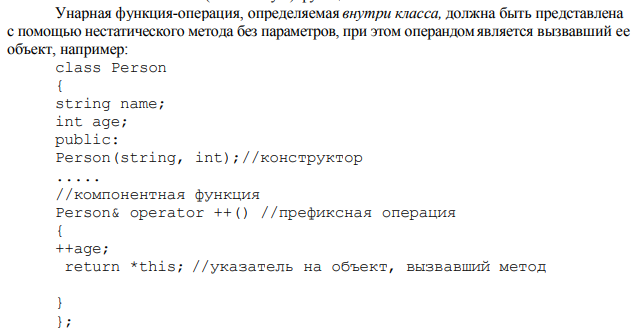


2. 



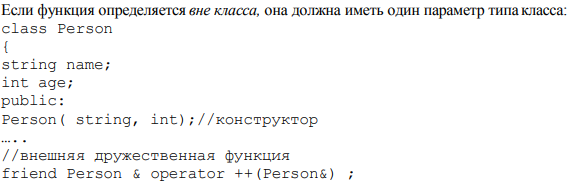
3. 

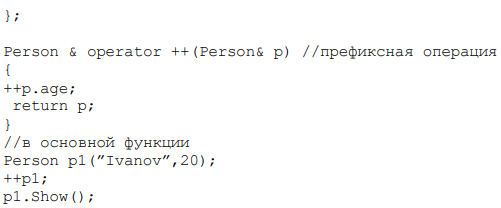


4.

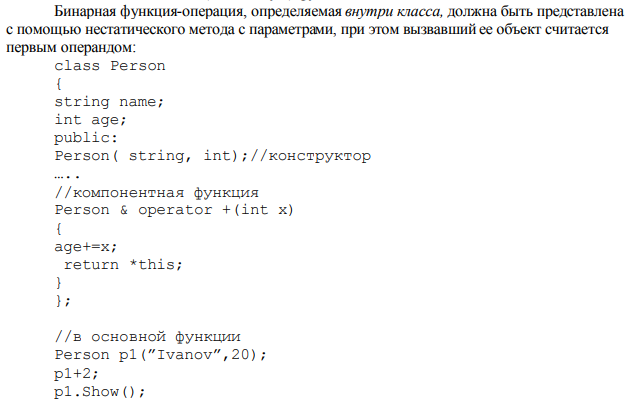


16

5.

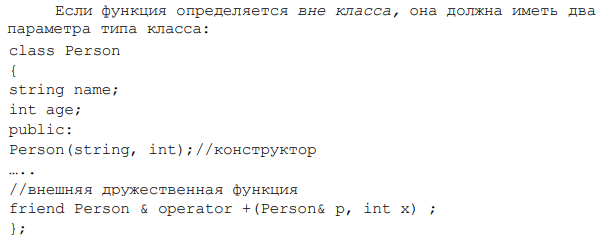




6.



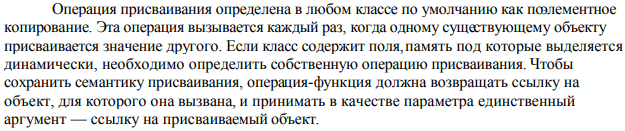
17

7. 



8. 

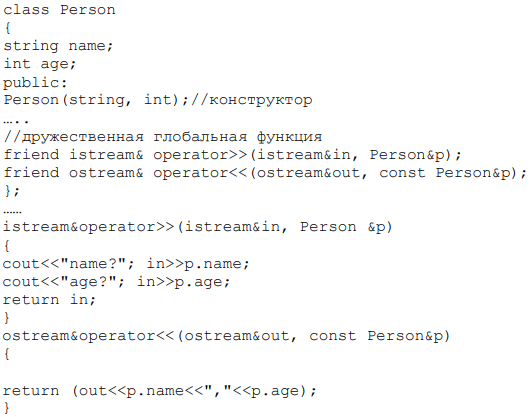


9. 

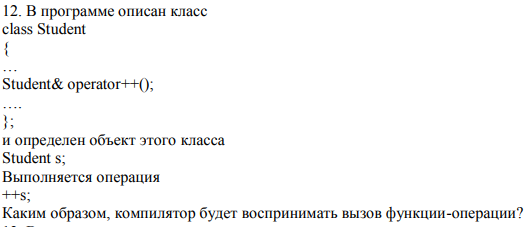


10. Ссылку на объект, для которого операция была вызвана.

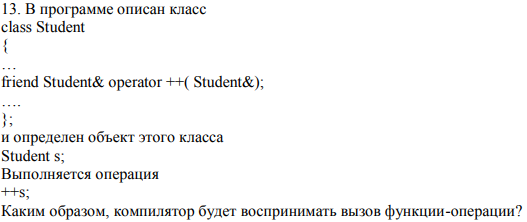




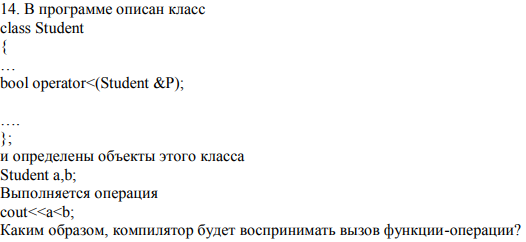
18



12. Будет осуществлён вызов функции перегрузки оператора префиксного инкремента.

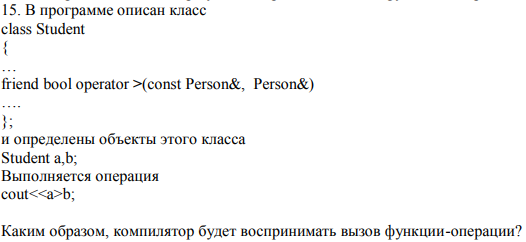


13. Будет осуществлён вызов функции перегрузки оператора постфиксного инкремента.



14. Будет осуществлён вызов перегруженного метода операции сравнения (<), а затем будет выведен результат сравнения (1 или 0).

19



15. Будет осуществлён вызов перегруженного метода операции сравнения (>), а затем будет выведен результат сравнения (1 или 0).