Видео: <https://youtu.be/7pXK7qzIpCY>

Экзаменационное задание по дисциплине «*Объектно-ориентированное программирование с использованием языка C++*»

Написать программу для проведения экзамена по дисциплине «*Объектно-ориентированное программирование с использованием языка C++*» ☺

1. Список вопросов, которые задаются студенту, находятся в файле **questions.txt** в папке с программой. Все вопросы разбиты на блоки. Каждый блок вопросов отделен друг от друга пустой строкой. Количество блоков вопросов, как и количество вопросов в каждом блоке заранее не определено. Вопросы в пределах блока нумеруются по порядку (пример вопросов см. в приложении, а также в файле questions.txt)
2. При запуске приложения необходимо открыть файл, считать все вопросы и сформировать блоки вопросов.
3. У студента запрашивается ФИО, группа.
4. Студенту задается столько вопросов, сколько блоков вопросов в файле questions.txt. Например, в приложении присутствует **три** блока вопросов, соответственно студенту задается три случайных вопроса в пределах каждого блока.
5. У студента есть возможность заменить вопрос, если он не знает ответ. При этом количество замен не должно превышать количества блоков вопросов. Смена вопроса возможно только в пределах данного блока.
6. Первая замена вопроса не ведет к штрафу, последующая – снимается один балл.
7. После готовности студента дать ответ на вопрос – программа предлагает студенту ввести письменный ответ. Студент последовательно отвечает на первый, второй и т.д. вопросы.
8. После ответа на последний вопрос экзамен считается сданным и результаты экзамена – вопросы и ответы записываются в файл **results.txt**.

Пример вывода программы (файл resulst.txt):

Студент: Иванов Иван Иванович

Группа: ПП-16-о1

Количество заданных вопросов: 5

--------------------------------------------------------------

Блок 1.

-------

Вопрос

3. Чем класс отличается от структуры? Можно ли использовать ключевые слова public и private в структуре?

\*\*\*\*\*\*\*\*\*\*\*\*

Ответ: В структурах все поля открытые по умолчанию, а в классах – закрытые. В структуре, как и в классе можно использовать private, protected и public.

--------------------------------------------------------------

Блок 2.

-------

Вопрос

1. Какие виды наследования возможны в С++?

Ответ: Студент не знает ответа на вопрос. Первая замена.

-------

Замена вопроса

16. Наследуются ли виртуальные функции?

\*\*\*\*\*\*\*\*\*\*\*\*

Ответ: Да, наследуется.

--------------------------------------------------------------

Блок 3.

-------

Вопрос

1. Объясните, для чего нужны строковые потоки.

Ответ: Студент не знает ответа на вопрос. Вторая замена.

----------

Замена вопроса

12. Каким образом ввести строку с пробелами?

\*\*\*\*\*\*\*\*\*\*\*\*

Ответ: Используя метод getline(char\* buff) или get(char\* buff, int size, char delim).

Приложение

1. Что такое класс? Чем отличается класс от объекта?
2. Допускается ли передавать объекты в качестве параметров, и какими способами? А возвращать как результат?
3. Чем класс отличается от структуры? Можно ли использовать ключевые слова public и private в структуре?
4. Объясните принцип инкапсуляции. Обязательно ли делать поля класса приватными?
5. Для чего используются ключевые слова public и private? Существуют ли ограничения на использование public и private в классе?
6. Что такое метод? Как вызывается метод? Что такое параметры значения по умолчанию?
7. Что такое наследование?
8. Как определить метод непосредственно внутри класса и вне класса? Чем эти определения отличаются?
9. Что обозначается ключевым словом this?
10. Зачем нужны константные методы? Чем отличается определение константного метода от обычного?
11. Объясните принцип полиморфизма.
12. Можно ли перегружать операции для встроенных типов данных?
13. Перечислите особенности перегрузки операций через методы класса. Чем отличается перегрузка через функции от перегрузки через методы класса?
14. Какой результат должны возвращать операции с присваиванием?
15. Какие операции разрешается перегружать только как методы класса?
16. Дайте определение дружественной функции. Как объявляется дружественная функция? А как определяется?
17. Дайте определение конструктора. Каково назначение конструктора? Перечислите отличия конструктора от метода.
18. Сколько конструкторов может быть в классе? Допускается ли перегрузка конструкторов? Какие виды конструкторов создаются по умолчанию?
19. Может ли конструктор быть приватным? Какие последствия влечет за собой объявление конструктора приватным?
20. Приведите несколько случаев, когда конструктор вызывается неявно. Как запретить неявное преобразование типа?
21. Для чего служит ключевое слово explicit?
22. Сколько операндов имеет операция индексирования []? Что она должна возвращать?
23. Для чего нужны статические поля в классе? Как они определяются? Чем отличаются от обычных полей?
24. Дайте определение итератора. Что играет роль итератора для массивов С++?
25. Что такое деструктор? Может ли деструктор иметь параметры?
26. Что такое перегрузка? Чем отличается перегрузка от переопределения?
27. Что такое конструктор копирования? Чем отличается конструктор копирования от обычного конструктора? Когда необходимо применять?
28. Что такое односвязный список?
29. Чем двусвязный список отличается от односвязного?
30. Что такое очередь?
31. Чем кольцевая очередь отличается от очереди с приоритетом?
32. Что такое стек?
33. Что такое бинарное дерево поиска?
34. Что такое исключение? Назовите ключевые слова С++, которые используются для обработки исключений.
35. Каким образом генерируется исключение? Как перехватить исключение?
36. Какого типа может быть исключение?
37. Сколько параметров разрешается писать в секции try…catch?
38. Напишите конструкцию, которая позволяет перехватить любое исключение.
39. Могут ли блоки try…catch быть вложенными?
40. Перечислите возможные способы выхода из блока try…catch.
41. Каким образом исключение «передать дальше»?
42. Сколько секций catch должно быть задано в блоке try?
43. Существуют ли стандартные исключения? Назовите два-три типа стандартных исключений.
44. Какие виды наследования возможны в С++?
45. Чем отличается модификатор доступа protected от модификаторов private и public?
46. Чем открытое наследование отличается от закрытого и защищенного?
47. Каков порядок вызова конструкторов? А деструкторов?
48. Можно ли в производном классе объявлять новые поля? А методы? Если имя нового поля совпадает с именем унаследованного, то что тогда?
49. Что происходит, если имя метода-наследника совпадает с именем базового метода?
50. Поястните возможно ли приведение типов от базового класса к производному и обратно?
51. Поясните проблему множественного наследования? Какие возможны способы решения проблемы множественного наследования.
52. Объясните, зачем нужны виртуальные функции.
53. Чем «раннее» связывание отличается от «позднего»?
54. Какие два вида полиморфизма реализованы в С++?
55. Дайте определение полиморфного класса.
56. Наследуются ли виртуальные функции?
57. Может ли конструктор быть виртуальным? А деструктор?
58. Как объявляется «чистая» виртуальная функция? Чем отличается от просто виртуальной функции?
59. Дайте определение абстрактного класса.
60. Перечислите назначение наследования.
61. Поясните как вызвать метод родительского класса если в наследнике также существует метод с такой же сигнатурой?
62. Для чего используются пространства имен?
63. Могут ли пространства имен быть вложенными?
64. Для чего применяются псевдонимы пространства имен
65. В чем состоит проблема глобальных переменных?
66. Чем отличается статический метод от обычного?
67. Какие методы класса не могут быть статическими?
68. Какие применения статических полей вы можете привести? А каким образом применяются статические методы?
69. Приведите структуру и принцип действия паттерна Singleton.
70. Для чего предназначены шаблоны?
71. Объясните назначение ключевого слова typename.
72. Можно ли параметрам шаблона присваивать значения по умолчанию?
73. Зачем нужны умные указатели?
74. Что такое «стратегия владения» умного указателя? Сколько стратегий владения вы знаете?
75. Какие умные указатели реализован в стандартной библиотеке С++?
76. Что такое «функтор»?
77. Объясните, чем отличается множественное наследование от простого?
78. Что такое «поток» — дайте определение.
79. Какие классы ввода/вывода реализованы в С++ и чем они отличаются?
80. Перечислите стандартные потоки и ввода-вывод объясните их назначение
81. Объясните, для чего нужны строковые потоки.
82. С помощью каких функций выполняется работа со строковыми потоками?
83. Объясните, в чем заключается различие между текстовым и двоичным файлом.
84. Объясните, что означает «открыть» файл и «закрыть» файл?
85. Перечислите режимы открытия файла. Чем отличается режим ios::in от режима ios::out?
86. В каких случаях необходимо следить за ситуацией «конец файла»? Каким способом это делается?
87. Можно ли текстовый файл открыть как двоичный? А двоичный — как текстовый?
88. Какие функции ввода/вывода используются для обмена с текстовыми файлами?
89. Перечислите функции ввода/вывода для работы с двоичными файлами.
90. Объясните назначение методов seekg() и tellg().
91. В чем преимущества объектно-ориентированной библитеки по работе с файлами по сравнению с процедурной?
92. Каким образом ввести строку с пробелами?
93. Перечислите режимы открытия объектно-ориентированных файловых потоков. Каким образом комбинируются режимы открытия файлоавых потоков?
94. Обязательно ли закрывать файл, связанный с объектно-ориентированным файловым потоком? А открывать?
95. Каким образом открыть файловый поток для чтения и записи одновременно?
96. Как открыть файловый поток для дозаписи?
97. Можно ли вывести значение переменной в двоичном виде и как это сделать?
98. Покажите, каким образом можно выполнить перегрузку операций ввода/вывода для нового типа данных.
99. Перечислите последовательные контейнеры стандартной библиотеки. Чем они отличаются друг от друга?
100. Перечислите виды конструкторов, с помощью которых можно создавать вектор.
101. Можно ли инициализировать контейнер элементами массива? А элементами другого контейнера? Какими способами это можно сделать?
102. Какие методы реализованы в контейнере-векторе для доступа к элементам?
103. Отличается ли функция at() доступа по индексу от перегруженной операции индексирования и чем?
104. Чем метод size() отличается от метода capacity()? А в чем отличие этих методов от метода max\_size()?
105. Какие контейнеры называются ассоциативными и почему?
106. Чем контейнер map отличается от контейнера multimap?
107. Как используется pair<> в ассоциативных контейнерах?
108. Какими преимуществами обладает функция make\_pair() по сравнению с конструктором pair()?
109. Чем отличается метод c\_str() от мтода data() класса string?
110. Дайте определение итератора.
111. Что такое «начальный» итератор и «конечный» итератор? Какие методы, связанные с итераторами, обязательно включает каждый контейнер?
112. Чем константный итератор отличается от неконстантного?
113. Объясните, почему итераторы реализованы как вложенные классы в контейнерах.
114. Почему алгоритмы remove() не удаляют элементы из vector? Как реально удалить элементы из контейнера?
115. Какую функцию выполняет алгоритмы unique()?
116. Нужно ли сортировать ассоциативные контейнеры?
117. Какие алгоритмы предназначены для заполнения контейнера значениями?
118. Каким образом заполнить с помощью алгоритма generate() последовательный контейнер?
119. Можно ли с помощью алгоритма for\_each() изменить элементы контейнера
120. Что такое lambda выражение? Чем отличается от функции?
121. Что такое предикат?