Вопросы по дисциплине: Основы программирования на с++

1. Почему в языке С++ определена строгая типизация данных, используемых в программе?
2. Как определяются границы диапазона базового типа в зависимости от выделяемой под этот тип памяти?
3. С какой целью в С++ определен тип void?
4. Какой объем памяти выделяется под переменную типа void? Какие значения может принимать переменная типа void?
5. Почему наблюдается асимметрия значений границ диапазонов целочисленных типов?
6. Чему будет равно значение операции инкремента для максимального числа в целочисленном типе? А каков результат декремента для минимального значения в таком же типе?
7. Почему запись целых чисел нельзя начинать с незначащих нулей?
8. Каким образом представлено число ноль в вещественных типах?
9. Почему в С++ символьный тип считается подмножеством целочисленного типа?
10. Каким образом можно инициализировать переменную перечисляемого типа?
11. При преобразовании целого со знаком к целому без знака всегда ли будет получено исходное числовое значение? Ответ обоснуйте.
12. Чем отличается механизм вызова встраиваемой и обычной пользовательских функций?
13. На каком этапе выполнения программы происходит встраивание кода подставляемой функции?
14. Почему не рекомендуется встраивать функции с большим кодом?
15. Как будет выполняться программа, если размер кода подставляемой функции превосходит допустимый свободный размер памяти?
16. С какой целью в программировании реализован полиморфизм функций?
17. Каким образом компилятор определяет, какую из перегруженных функций необходимо вызвать в программном коде?
18. Могут ли перегруженные функции возвращать результат одного типа? Ответ обоснуйте.
19. Приведите примеры рекурсивных объектов и явлений. Обоснуйте проявление рекурсивности.
20. Почему при правильной организации рекурсивные вызовы не зацикливаются?

21. Почему не отождествляются совпадающие идентификаторы при многократных рекурсивных вызовах?

22. Почему рекурсивные обращения завершаются в порядке, обратном вызовам этих обращений?

23. Чем ограничено при выполнении программы количество рекурсивных вызовов?

24. Какой из методов в программировании является более эффективным – рекурсивный или итерационный?

25. Почему указатель не может существовать как самостоятельный тип?

26. С какой целью в программе может быть использован указатель типа void?

27. Что будет являться результатом разыменования указателя типа void без приведения типов?

28. Как изменится значение указателя после применения к нему операции инкремента (декремента)?

29. Почему для указателей определены сложение и вычитание только с целыми константами?

30. В чем отличие указателя на константу от указателя-константы?

31. Два указателя разных типов указывают на одно и то же место в памяти. Опишите результаты операций разыменования и взятия адреса с такими указателям.

32. Если объект занимает в памяти несколько байт, то какой адрес является значением указателя на этот объект?

33. Каким образом при разыменовании указателей становится известно, сколько байт памяти доступно?

34. Почему запрещены арифметические операции над указателями на функции?

35. Почему в описании указателя на функцию необходимы круглые скобки при имени указателя?

36. Может ли функция возвращать значение типа указатель? Если да, то как объявляется прототип такой функции?

37. Могут ли параметрами функции быть указатели на объекты? Если да, то как происходит передача фактических параметров при вызове функции?

38. Могут ли параметрами функции быть указатели на функции? Если да, то как происходит передача фактических параметров при вызове функции?

39. Как понимается следующее объявление: float *(*func)(int(*pf)(char),float); ?

40. В чем отличие результатов вызова функции через указатель с последующим разыменованием указателя и без разыменования указателя?

41. С какой целью в программах используют указатели на указатели?

42. Что будет являться результатом однократной операции разыменования указателя, реализующего многочисленное перенаправление?

43. Для чего в программе необходима инициализация указателя перед первым его использованием?

44. С какой целью в прототипах функций с переменным числом параметров должны быть указаны обязательные параметры?

45. Как в функции с переменным числом параметров осуществляется доступ к списку неизвестных параметров?

46. Почему для доступа к списку неизвестных параметров достаточно знать адрес хотя бы одного обязательного параметра?

47. Почему ошибки, связанные с некорректным использованием указателей, относятся к наиболее трудноустранимым?

48. Почему в C++ не выполняется операция прямого присваивания значения строке?

49. Почему символ и строка, состоящая из одного символа, занимают разный объем памяти?

50. Почему в функции scanf("%s",string); не указывается обращение к переменной по адресу?

51. Допустима ли операция сравнения над символами? Если да, то каким образом определены отношения "больше" и "меньше"?

52. Какая из функций, gets или puts, заносит в поток управляющий символ '\n' и с какой целью?

53. Можно ли выполнить присваивание символьной переменной числового значения? Почему?

54. В чем различия результатов вывода символьной переменной со спецификаторами %d и %c?

55. Что будет являться результатом работы функции побайтового копирования строк, если длина строки-источника превосходит допустимый размер строки-приемника?

56. Что будет являться результатом работы функции побайтового копирования строк, если длина строки-источника меньше размера строки-приемника?

57. Почему при сравнении строк важен регистр символов?

58. Как сравниваются строки разной длины?

59. Какие возможны последствия при обращении к неинициализированной строке?

60. Почему функция изменения регистра символов строки может некорректно работать с кириллицей?

61. Почему обращения к строке через ее имя и через указатель эквивалентны?

62. Почему в качестве параметра функции передается адрес строки, а не сама строка символов?

63. Возможно ли применение операций инкремента и декремента к указателю на строку? Если да, то что будет адресовать полученный указатель?

64. Почему при формировании строки без использования стандартных функций необходимо дописывать символ конца строки? Почему этого не требуется при считывании строк с клавиатуры?

65. Какие возможны ошибки в программе при некорректной работе со строками?

66. Для защиты строки от изменения объявляется указатель на константу или указатель-константа? Почему?

67. Почему в программе на C++ необходимо, чтобы был известен размер массива?

68. Можно ли выполнить прямое присваивание массивов объявленных так: int x[10], y[10];?

69. Когда, с какой целью и почему возможно объявление безразмерных массивов?

70. В чем отличие обращения к элементам массива с помощью индексированного имени и посредством арифметики с указателями?

71. Может ли значение элемента массива использоваться в качестве индекса другого элемента массива?

72. Эквивалентны ли для массива mas следующие обращения и почему: mas и &mas[0]?

73. Какие ограничения распространяются на тип массива?

74. Каким образом можно определить объем памяти, выделяемой под массив?

75. Каким образом можно составить выражение для генерации массива случайными целыми числами на заданном промежутке?

76. Какие классы задач предполагают изменение значений элементов массива?

77. Какие классы задач предполагают только изменение порядка следования элементов в массиве?

78. Каким образом можно выполнять обход массива?

79. Почему в алгоритме циклического сдвига элементов массива важен порядок смещения элементов?

80. Чем различаются алгоритмы поиска первого и последнего минимального (максимального) элемента в массиве?

81. Почему существует большое количество алгоритмов сортировок?

82. С какой целью используются простые сортировки, если они характеризуются малой эффективностью?

83. Чем отличается принцип сортировки по неубыванию (невозрастанию) от сортировки по возрастанию (убыванию)?

84. На каких наборах исходных данных проявляется эффективность алгоритмов простых сортировок по сравнению друг с другом?

85. В чем заключается улучшение метода шейкер-сортировки по сравнению с пузырьковой сортировкой?

86. Почему в программе на C++ при объявлении двумерного массива необходимо, чтобы был известен размер по каждому измерению массива?

87. Можно ли выполнить прямое присваивание двумерных массивов?

88. Когда, с какой целью и почему возможно объявление безразмерных массивов? С одним безразмерным измерением?

89. В чем отличие обращения к элементам двумерного массива с помощью индексированного имени и посредством арифметики с указателями?

90. Эквивалентны ли для массива mas следующие обращения и почему: mas и &mas[0][0]?

91. Возможно ли в двумерном массиве mas обращение к элементу &mas[0]? Почему?

92. Приведите возможные обращения к элементу двумерного массива, аналогичные обращению mas[i][j].

93. Какие ограничения распространяются на тип массива?

94. Каким образом можно определить объем памяти, выделяемой под двумерный массив?

95. В чем принципиальное отличие задач сортировок двумерных и одномерных массивов?

96. Каким образом оформляется прототип функции, чтобы изменения, выполненные с элементами массива, были сохранены после завершения работы функции?

97. Приведите возможные обращения к элементу трехмерного массива, аналогичные обращению mas[i][j][k].

98. В чем причина неудобства использования массивов слишком больших измерений в программах?

99. При решении каких прикладных задач используются многомерные массивы? Отдельно приведите примеры для массивов с измерением два и более.

100. В чем принципиальное отличие типов массив и структура?

101. Как располагаются в памяти элементы структуры?

102. Почему размер структуры не всегда совпадает с суммарным размером ее полей?

103. Для моделирования каких данных целесообразно использовать структуры?

104. Какими способами можно обратиться к данным структуры?

105. В чем отличие прямого и косвенного доступа к полям структуры?

106. Всегда ли возможно выполнить напрямую операцию присваивания значений объектов структуры с одинаковым набором полей?

107. При каком объявлении структурных объектов возможно выполнить напрямую операцию присваивания значений объектов структуры?

108. Для моделирования каких данных целесообразно использовать массив структур?

109. Какие данные о структуре содержит указатель на эту структуру?

110. Какие ограничения накладываются на тип элемента структуры?

111. Возможно ли в качестве типа элемента структуры использовать указатель на другую структуру?

112. Как выполняется инициализация указателя на структуру?

113. Как выполняются операции инкремента и декремента над указателями на структуры?

114. Какими способами можно обратиться к данным структуры, используя указатели?

115. Каким образом необходимо передать структуру в качестве параметра функции, чтобы сохранить изменения, совершаемые функцией с данной структурой?

116. Что возвращается в качестве значения функции, тип которой объявлен как структура?

117. Что возвращается в качестве значения функции, тип которой объявлен как указатель на структуру?

118. В чем принципиальное отличие размещения в памяти элементов структуры и объединения?

119. Каким образом определяется размер объединения?

120. Какова цель использования объединений в программировании?

121. Какую информацию об объединении содержит указатель на это объединение?

122. Какое значение будет храниться в объединении, если будут проинициализированы все поля?

123. Какие существуют способы обращения к элементам объединения?

124. В чем отличия различных способов обращения к элементам объединения?

125. Для моделирования каких данных целесообразно использовать массив объединений?

126. Как и с какой целью объявляются переменные с изменяемой структурой?

127. Какого типа объявляются битовые поля?

128. Почему нельзя объявить битовые поля вне структуры или объединения?

129. Как осуществляется адресация битовых полей?

130. От чего зависит порядок размещения в памяти битовых полей?

131. Возможно ли объявление массива битовых полей? Почему?

132. Возможно ли объявить безымянное битовое поле? Если да, то с какой целью оно используется?

133. Возможно ли объявить битовое поле нулевой длины? Если да, то с какой целью оно используется?

134. Существуют ли ограничения на длину битового поля?

135. С какой целью используются объединения с битовыми полями?

136. Какие существуют способы обращения к битовым полям?

137. В чем отличия текстовых и двоичных файлов с точки зрения представления данных?

138. Почему поток ввода-вывода не зависит от конкретного устройства?

139. Для чего необходима буферизация при потоковом и форматированном вводе-выводе?

140. С какой целью предусмотрены режимы открытия файлов и почему их такое многообразие?

141. Каковы могут быть причины ошибок при открытии файлов?

142. Какие значения возвращает функция открытия файла в зависимости от результата?

143. Каким образом в программе происходит проверка достижения конца файла?

144. Может ли один и тот же указатель на файл одновременно связан с несколькими файлами? Почему?

145. Может ли один и тот же файл одновременно быть открыт для чтения и для записи?

146. Можно ли один и тот же файл открыть несколько раз, не закрывая после каждого открытия?

147. Сохранится ли информация в файле, если его не закрыть в программе с помощью функции?

148. В чем основные отличия в организации символьного, строкового, блокового и форматированного ввода-ввода в файлы?

149. Почему ввод-вывод в файлы в потоковом режиме аналогичен работе с другими устройствами ввода-вывода?

150. С какой целью предусмотрены режимы открытия файлов и почему их такое многообразие?

151. Каковы могут быть причины ошибок при открытии файлов в потоковом режиме?

152. Какие значения возвращает функция открытия файла в потоковом режиме в зависимости от результата?

153. Каким образом в программе происходит проверка достижения конца файла?

154. Может ли один и тот же файл одновременно быть открыт для чтения и для записи?

155. Можно ли один и тот же файл открыть несколько раз, не закрывая после каждого открытия?

156. Сохранится ли информация в файле, если его не закрыть в программе с помощью функции? Обоснуйте ответ.

157. В чем принципиальное отличие ввода-вывода в файлы при потоковом и стандартном режимах?

158. Почему передача данных при файловом вводе-выводе осуществляется побайтно?

159. Почему при обмене данными через потоки файлы отождествляются с внешними устройствами?

160. Возможно ли расположить указатель в файле перед началом файла? Обоснуйте ответ.

161. Возможно ли расположить указатель в файле после символа конца файла? Обоснуйте ответ.

162. По какой причине для потоков, открытых в режиме преобразования типов, применение fseek является ограниченным?

163. Как изменится файл, если произвести запись данных в середину файла при буферизированном вводе-выводе?

164. В чем преимущества и недостатки каждого из способов организации рабочей памяти (в виде массива или другого файла) при редактировании файла? Предусмотрены ли флаги форматирования для вводимых в файл значений? Почему?

165. С какой целью выделены как отдельные потоки ofstream, ifstream, fstream?

166. В чем отличие выполнения функции get без параметров и с тремя параметрами?

167. В чем отличие выполнения функции flags без параметров и с параметром?

168. Назовите возможные причины ошибок, возникающих при открытии файлов для чтения и для записи.

169. Можно ли один и тот же файл одновременно открыть для записи и для чтения? Если да, то где будет находиться файловый указатель?

170. Для чего используется динамическая память в программировании?

171. Какая область памяти выделяется под размещение динамических данных?

172. Как долго хранятся данные в динамической памяти?

173. Какие возможны варианты доступа к динамической памяти?

174. Что возвращает операция выделения динамической памяти в случае успешного выполнения?

175. Что возвращает операция выделения динамической памяти, если участок требуемого размера не может быть выделен?

176. Почему тип функций выделения динамической памяти определен как *void?

177. Почему при завершении работы с динамической памятью ее необходимо освободить? Какие могут быть последствия для работы программы, если не освобождать динамическую память?

178. Существуют ли ограничения на данные при применении к ним операции или функции освобождения динамической памяти?

179. В каких ситуациях в программировании целесообразно использовать динамические массивы?

180. Что будет возвращено при попытке объявить динамический массив недопустимо большого размера?

181. Как размещаются в памяти элементы одномерного динамического массива?

182. С какой целью используется первая пара скобок при объявлении (Тип*) malloc(N* sizeof(Тип))?

183. С какой целью выполняется явное преобразование типов значений функций malloc (calloc) при объявлении массивов?

184. Какими способами можно обратиться к элементам одномерного динамического массива?

185. С какой целью используются квадратные скобки в операции освобождения динамической памяти, выделенной под массив: delete [] mass;?

186. В чем сходство и отличие одномерных и двумерных динамических массивов?

187. Как размещаются в памяти элементы двумерного динамического массива?

188. Что является значением двойного указателя при объявлении двумерных динамических массивов?

189. Как выделяется память для двумерных динамических массивов?

190. Какими способами можно обратиться к элементам двумерного динамического массива?

191. Назовите порядок освобождения памяти, выделенной под двумерный динамический массив.

192. Какая область динамической памяти, выделенной под двумерный динамический массив, будет освобождена, если только применить операцию: delete [] mass;?

193. Какова цель использования тройных указателей в программах? Всегда ли эффективно использование динамической памяти в программах?

194. Почему ошибки при работе с динамической памятью относят к опасным?

195. Опишите последствия использования в программах неинициализированных указателей.

196. Почему в программах не следует оставлять "висячие" указатели?

197. Почему выделенную ранее динамическую память следует освобождать после использования?

198. К каким последствиям в работе программы приводит попытка освободить динамическую память, не выделенную ранее?

199. Является ли рекурсия универсальным способом решения задач? Ответ обоснуйте.

200. Почему к нулевому указателю нельзя применить операцию разыменования?

201. Почему рекурсию нельзя рассматривать как универсальный метод решения задач?

202. Какие используются способы представления двумерных динамических массивов?

203. Назовите порядок выделения и освобождения памяти под двумерный динамический массив. Укажите различия для разных способов представления двумерных динамических массивов.

204. Почему в программах размер памяти под статические переменные должен быть определен на этапе компиляции?

205. Чем можно объяснить многообразие алгоритмов поиска в линейных структурах?

206. В чем преимущества поиска с барьером по сравнению с последовательным поиском?

207. Нахождение какого по порядку элемента в линейном множестве (первого, последнего) гарантирует алгоритм прямого поиска? Как в этом случае должен быть выполнен просмотр?

208. Нахождение какого по порядку элемента в линейном множестве (первого, последнего) гарантирует алгоритм бинарного поиска? Ответ обоснуйте.

209. Как трудоемкость алгоритма бинарного поиска на дискретном множестве зависит от мощности множества?

210. Почему время выполнения алгоритма бинарного поиска на вещественном множестве не зависит от количества элементов?

211. Приведите пример входных данных для реализации эффективного метода прямого поиска подстроки в строке.