МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

(ФГБОУ ВО «ВятГУ»)

Факультет автоматики и вычислительной техники

Кафедра ЭВМ

Отчёт

Лабораторная работа № 2 по дисциплине

«Технология программирования»

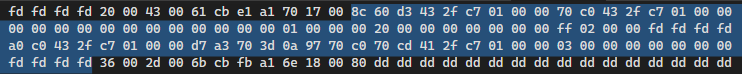
«Исследование состояния динамический памяти при работе с элементарными структурами данных»

Выполнил студент группы ИВТб-2301-04-00 / Жеребцов К. А.

Проверил преподаватель / Долженкова М. Л.

Киров 2022

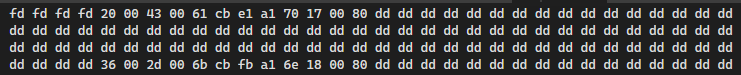
1. **Цель:** изучить состояния динамический памяти при работе с элементарными структурами данных.
2. **Задание:** написать программу для работы с динамической структурой данных очередь, содержащей в каждом элементе массив символов и вещественное число.
3. **Работа программы:**
   1. Занесение элемента:



* Индикатор начала – 8С16 = 1000 1**1**00 (Выделенный бит – флаг занятости участка)
* Указатель на предыдущий занятый элемент
* Указатель на следующий занятый элемент
* Указатель на имя файла подкачки
* Номер строки в файле подкачки
* Тип участка памяти (01 – пользовательский тип)
* Размер выделенного участка памяти
* Количество обращений
* Индикатор начала пользовательского участка памяти
* Пользовательская структура из 32 байт (1 – указатель на массив char, 2 – число типа float, 3 – указатель на следующий элемент, 4 – размер массива char)
* Индикатор конца пользовательского участка памяти
  1. Удаление элемента
* очистка массива символов из 3 элементов



* очистка структуры



Произошло освобождение области памяти после удаления элемента из очереди, флаг занятости сменился на 0 – 8016 = 1000 0**0**00

1. **Листинг программы:**
2. #include <iostream>
3. #include <cstdlib>
4. #include <string>
5. using namespace std;
6. template<class T>
7. T readnumber(int min, int max, T type)
8. {
9. bool flag = true;
10. do
11. {
12. cin >> type;
13. if (!cin.good() || cin.get() != '\n')
14. {
15. system("cls");
16. cout << "Ошибка! Повторите ввод: \n";
17. cin.clear();
18. cin.ignore(255, '\n');
19. }
20. else if (type < min || type > max)
21. {
22. system("cls");
23. cout << "Вы ввели не верное число. \n";
24. }
25. else flag = false;
26. } while (flag == true);
27. return type;
28. }
29. char readchar()
30. {
31. bool flag = true;
32. char symb;
33. do
34. {
35. cin >> symb;
36. if (!cin.good() || cin.get() != '\n')
37. {
38. system("cls");
39. cout << "Ошибка! Повторите ввод: \n";
40. cin.clear();
41. cin.ignore(255, '\n');
42. }
43. else flag = false;
44. } while (flag == true);
46. return symb;
47. }
48. struct node {
49. char\* str;
50. double f;
51. node\* next;
52. int size;
53. };
54. int main()
55. {
56. int num;
57. node\* Q\_first = NULL;
58. node\* Q;
59. node\* Q1;
60. int nume = 0;
61. setlocale(LC\_ALL, "ru");
62. do {
64. cout << endl;
65. cout << ("0. Выход \n");
66. cout << ("1. Добавить элемент \n");
67. cout << ("2. Вывод очереди \n");
68. cout << ("3. Удалить улемент \n");
69. cout << ("4. Очистить очередь \n");
70. cout << ("Выбирите номер команды: \n");
72. num = readnumber(0, 4, 1);
73. switch (num)
74. {
75. case 1:
76. {
77. if (nume == 0)
78. {
79. Q = (node\*)calloc(1, sizeof(node));
80. cout << ("Введите количество элементов массивы символов: \n");
81. Q->size = readnumber(1, 100, 1);
82. Q->str = (char\*)calloc(Q->size, sizeof(char));
84. for (int i = 0; i < Q->size; i++)
85. {
86. cout << "Введите " << i << " элемент массива: ";
87. Q->str[i] = readchar();
88. }
89. cout << "Введите число: ";
90. Q->f = readnumber(-10000000, 10000000, 1.0);
91. Q->next = Q;
92. Q\_first = Q;
93. nume++;
94. break;
95. }
96. else
97. {
98. Q1 = (node\*)calloc(1, sizeof(node));
99. cout << ("Введите количество элементов массивы символов: \n");
100. Q1->size = readnumber(1, 100, 1);
101. Q1->str = (char\*)calloc(Q1->size, sizeof(char));
102. for (int i = 0; i < Q1->size; i++)
103. {
104. cout << "Введите " << i << " элемент массива: ";
105. Q1->str[i] = readchar();
106. }
107. cout << "Введите число: ";
108. Q1->f = readnumber(-10000000, 10000000, 1.0);
109. Q1->next = Q\_first;
110. Q = Q\_first;
111. for (int i = 1; i < nume; i++)
112. {
113. Q = Q->next;
114. }
115. Q->next = Q1;
116. nume++;
117. break;
118. }
119. }
120. case 2:
121. {
122. if (Q\_first != NULL)
123. {
124. Q = Q\_first;
125. for (int j = 0; j < nume; j++)
126. {
127. cout << "Массив символов " << j + 1 << " элемента очереди: ";
128. for (int i = 0; i < Q->size; i++)
129. {
130. cout << Q->str[i] << " ";
131. }
132. cout << "\n";
133. cout << "Вещественное число " << j + 1 << " элемента очереди: ";
134. cout << Q->f << "\n";
135. Q = Q->next;
136. }
137. }
138. else cout << "Очередь пуста \n";
139. break;
140. }
141. case 3:
142. {
143. if (Q\_first != NULL)
144. {
145. Q = Q\_first->next;
146. free(Q\_first->str);
147. free(Q\_first);
148. Q\_first = Q;
149. nume--;
150. if (nume == 0)
151. {
152. Q\_first = NULL;
153. }
154. cout << "Элемент удален \n";
155. }
156. else
157. cout << "Очередь пуста \n";
158. break;
159. }
160. case 4:
161. {
162. if (Q\_first != NULL)
163. {
164. for (int i = 1; i < nume; i++)
165. {
166. Q = Q\_first->next;
167. free(Q\_first->str);
168. free(Q\_first);
169. Q\_first = Q;
170. }
171. free(Q\_first->str);
172. free(Q\_first);
173. Q\_first = NULL;
174. nume = 0;
175. cout << ("Очередь очищена \n");
176. }
177. else cout << "Очередь пуста \n";
178. break;
179. }
180. case 0:
181. {
182. if (Q\_first != NULL)
183. {
184. for (int i = 1; i < nume; i++)
185. {
186. Q = Q\_first->next;
187. free(Q\_first->str);
188. free(Q\_first);
189. Q\_first = Q;
190. }
191. free(Q\_first->str);
192. free(Q\_first);
193. Q\_first = NULL;
194. nume = 0;
195. cout << ("Очередь очищена \n");
196. }
197. else cout << "Очередь пуста \n";
198. num = 0;
199. break;
200. }
201. }
202. } while (num != 0);
203. return(0);
204. }

**5. Вывод:** было разработано программа для работы со структурой данных очередь с использованием функции выделения (calloc) и освобождения (free) памяти. Освоена работа с дампом памяти в VS.