**Практическая работа №14**

**Алгоритмы сжатия и кодирования данных.**

**Автор: Николаев-Аксенов И. С.**

**Группа: ИКБО-20-19**

**Код программы:**

1. #include <iostream>
2. #include <vector>
3. #include <list>
4. #include <string>
5. #include <algorithm>
6. #include <clocale>
7. #include <map>
8. #include <iomanip>
9. #include <queue>
10. using namespace std;
11. map<char, string> codes;
12. map<char, int> freq;
14. struct MinHeapNode
15. {
16. char data;
17. int freq;
18. MinHeapNode\* left, \* right;
20. MinHeapNode(char data, int freq)
21. {
22. left = right = NULL;
23. this->data = data;
24. this->freq = freq;
25. }
26. };

29. struct compare
30. {
31. bool operator()(MinHeapNode\* l, MinHeapNode\* r)
32. {
33. return (l->freq > r->freq);
34. }
36. };


40. void storeCodes(struct MinHeapNode\* root, string str)
41. {
42. if (root == NULL)
43. return;
44. if (root->data != '$')
45. codes[root->data] = str;
46. storeCodes(root->left, str + "0");
47. storeCodes(root->right, str + "1");
48. }

51. priority\_queue<MinHeapNode\*, vector<MinHeapNode\*>, compare> minHeap;
52. template<typename T> void print\_queue(T& q,int size)
53. {
54. priority\_queue<MinHeapNode\*, vector<MinHeapNode\*>, compare> q1=q;
55. priority\_queue<MinHeapNode\*, vector<MinHeapNode\*>, compare> q2 = q;
57. cout << "Алфавит: ";
58. while(!q.empty())
59. {
60. if (q.top() != NULL)
61. cout << setw(4) << q.top()->data << " ";
63. q.pop();
64. }
66. cout << endl << "Количество вхождений: ";
67. while (!q1.empty())
68. {
69. if (q1.top() != NULL)
70. cout << setw(4) << q1.top()->freq << " ";
72. q1.pop();
73. }
75. cout << endl << "Вероятность: ";
76. while (!q2.empty())
77. {
78. if (q2.top() != NULL)
79. cout << setw(4) << float(q2.top()->freq)/size << " ";
81. q2.pop();
82. }
84. cout << '**\n**';
85. }
86. void HuffmanCodes(int size)
87. {
88. struct MinHeapNode\* left, \* right, \* top;
90. for (map<char, int>::iterator v = freq.begin(); v != freq.end(); v++)
91. minHeap.push(new MinHeapNode(v->first, v->second));
92. priority\_queue<MinHeapNode\*, vector<MinHeapNode\*>, compare> minHeap2=minHeap;
93. print\_queue(minHeap2,size);
95. while (minHeap.size() != 1)
96. {
97. left = minHeap.top();
98. minHeap.pop();
99. right = minHeap.top();
100. minHeap.pop();
101. top = new MinHeapNode('$', left->freq + right->freq);
102. top->left = left;
103. top->right = right;
104. minHeap.push(top);
105. }
106. storeCodes(minHeap.top(), "");
107. }
109. void calcFreq(string str, int n)
110. {
111. for (int i = 0; i < str.size(); i++)
112. freq[str[i]]++;
113. }
115. string haffman\_code(string input)
116. {
117. string encodedString;
118. for (auto i : input)
119. encodedString += codes[i];
120. return encodedString;
121. }
123. void print\_table(string str)
124. {
125. cout << "Алфавит: ";
126. for (auto item : freq)
127. cout << setw(4) << item.first << " ";
129. cout << endl << "Количество вхождений: ";
130. for (auto item : freq)
131. cout << setw(4) << item.second << " ";
133. cout << endl << "Вероятность: ";
134. for (auto item : freq) {
135. cout.setf(std::ios::fixed);
136. cout << setprecision(2) << float(item.second)/str.length() << " ";
137. }
139. cout << endl;
140. cout << endl;
141. }
143. void sviaz\_codov()
144. {
145. for (auto s : codes)
146. cout << s.first << ": " << s.second << endl;
147. }
149. int dec2bin(int num)
150. {
151. int bin = 0, k = 1;
152. while (num)
153. {
154. bin += (num % 2) \* k;
155. k \*= 10;
156. num /= 2;
157. }
158. return bin;
159. }
161. string ascii\_code(string input) {
162. string asci = "";
163. for (int i = 0; i < input.size(); ++i) {
164. asci += to\_string(dec2bin((int(input[i]))));
165. }
166. return asci;
167. }
169. void results(string a) {
170. cout << "Коды символов: " << endl; sviaz\_codov();
171. cout << "Код по Хаффману: " << haffman\_code(a) << endl;
172. cout << "Длина кода по алгоритму Хаффмана: " << haffman\_code(a).length() << endl;
173. cout << "Код по ASCII: " << ascii\_code(a) << endl;
174. cout << "Длина кода по ASCII: " << ascii\_code(a).length() << endl;
175. cout << "Дисперсия " << ((float)haffman\_code(a).length() / ascii\_code(a).length()) << endl;
176. }
178. int main()
179. {
180. setlocale(LC\_ALL, "Russian");
181. string str = "nikolaevaxenov ivan sergeevich";
182. calcFreq(str, str.length());
183. print\_table(str);
184. HuffmanCodes(str.length());
185. results(str);
186. return 0;
187. }

**Результат выполнения программы:**

