F,Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Колледж ВятГУ

**ОТЧЕТ**

**ПО ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЕ №4**

**«РАБОТА В ГРАФИЧЕСКОМ РЕЖИМЕ»**

Выполнил: студент учебной группы

ИСПк-203-52-00

Абрамовский Артём Александрович

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2022

1. **Цель работы:** освоить принципы работы в графическом режиме; получить базовые навыки взаимодействия с графическими примитивами.
2. **Вариант: 1**

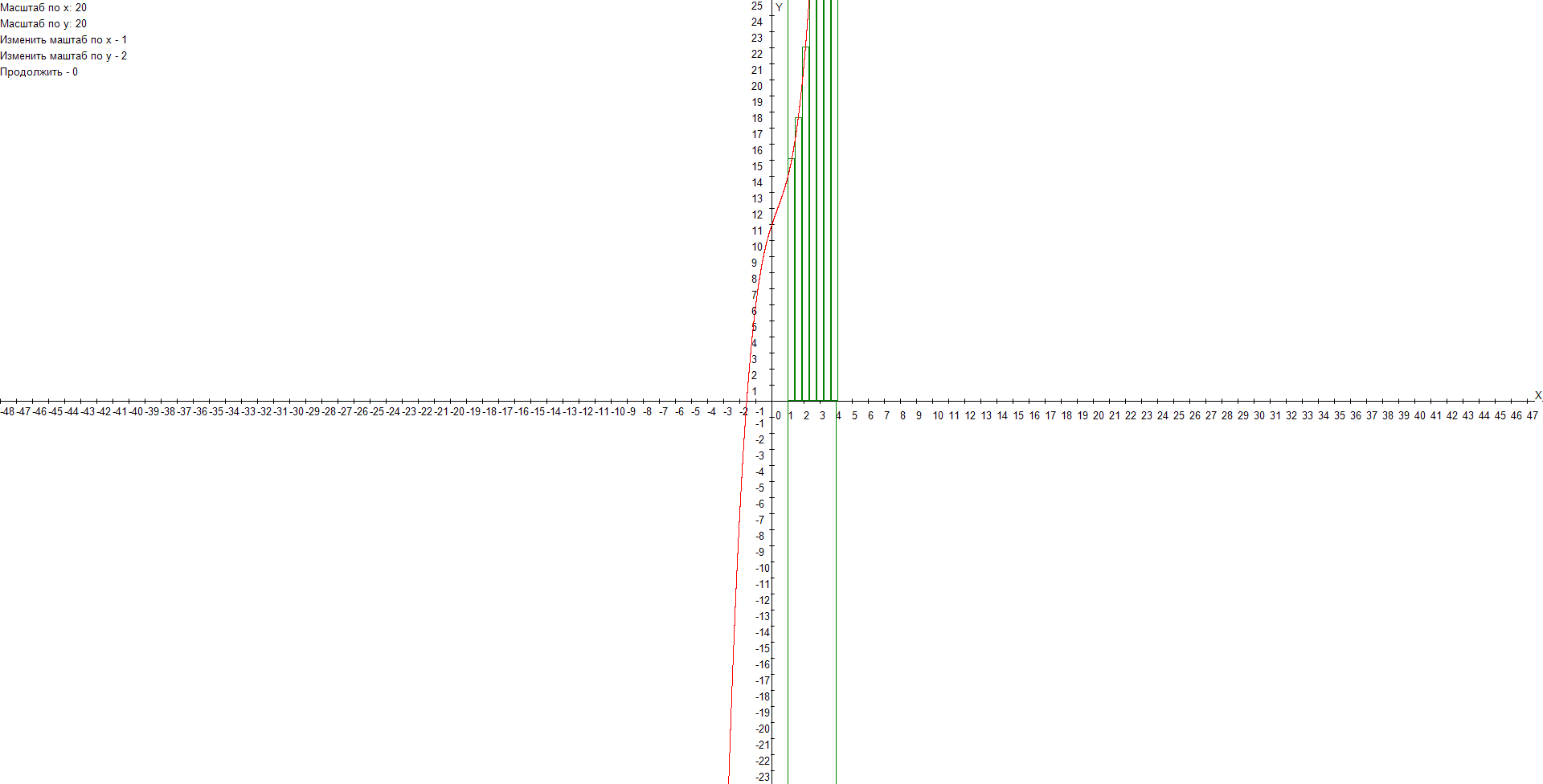
**Задание 1:** дополнить программу, реализованную в ходе предыдущей лабораторной работы, режимом визуализации.

**Задание 2:** предусмотреть возможность вывода кривой, ограничивающей фигуру, на координатную плоскость.

**Задание 3:** реализовать следующие возможности и элементы: масштабирование графика, подписи на осях, вывод информации о задании.

**Задание 4:** реализовать не менее двух возможностей из представленных: независимое масштабирование по осям, штриховка вычисляемой площади, визуализация численного расчета интеграла.

1. **Код программы:**
2. **uses** graphABC;
3. **var**
4. input: byte; m1, m2, p, u: real; n: integer;
5. **function** vib(**var** v: byte): byte;
6. **begin**
7. readln(v);
8. vib := v;
9. writeln;
10. **end**;
11. **function** fun(**var** x: real): real;
12. **var**
13. fo, s: real;
14. **begin**
15. fo :=power(x,3)+(-1)\*power(x,2)+(3)\*x+(11);
16. fun := fo;
17. **end**;
18. **function** fun1(**var** x: real): real;
19. **begin**
20. **var** fo: real;
21. fo := (power(x,4))/4-(power(x,3))/3+3\*((power(x,2))/2)+11\*x;
22. fun1 := fo;
23. **end**;
24. **function** graph(**var** s1, s2, h: real; ras: integer): integer;
25. **var**
26. x, mx, my: real; a, b, x0, y0, i: integer;
27. **begin**
28. MaximizeWindow;
29. clearwindow;
30. setpencolor(clblack);
31. a := -5;
32. b := 100;
33. x0 := windowwidth **div** 2;
34. y0 := windowheight **div** 2;
35. mx := m1;
36. my := m2;
37. line(0, y0, windowwidth, y0);
38. line(x0, 0, x0, windowheight);
39. **for** i := 1 **to** b **do**
40. **begin**
41. line(x0 + round(i \* mx), y0 - 3, x0 + round(i \* mx), y0 + 3);
42. line(x0 - round(i \* mx), y0 - 3, x0 - round(i \* mx), y0 + 3);
43. line(x0 - 3, y0 + round(i \* my), x0 + 3, y0 + round(i \* my));
44. line(x0 - 3, y0 - round(i \* my), x0 + 3, y0 - round(i \* my));
45. textout(x0 + round(i \* mx), y0 + 10, inttostr(i));
46. textout(x0 - round(i \* mx), y0 + 5, inttostr(-i));
47. textout(x0 - 25, y0 - round(i \* my), inttostr(i));
48. textout(x0 - 20, y0 + round(i \* my), inttostr(-i));
49. **end**;
50. textout(x0 + 5, y0 + 10, '0');
51. textout(windowwidth - 10, y0 - 15, 'X');
52. textout(x0 + 5, 10, 'Y');
53. x := a;
54. setpencolor(clgreen);
55. line(x0 + round(s1 \* mx), 0, x0 + round(s1 \* mx), windowheight);
56. setpencolor(clgreen);
57. line(x0 + round(s2 \* mx), 0, x0 + round(s2 \* mx), windowheight);
58. **var** l, w, k, e: real;
59. l := round((x0 + round(s2 \* mx) - x0 - round(s1 \* mx)) / n);
60. k := l;
61. p := (s2 - s1) / ras;
62. w := 0;
63. x := a;
64. **while** x <= b **do**
65. **begin**
66. **if** (x0 + round(s1 \* mx)) = (x0 + round(x \* mx)) **then**
67. **begin**
68. **if** y0 - round(fun(x) \* my) < y0 **then**
69. **begin**
70. setpencolor(clgreen);
71. u := p / 2 + x;
72. Rectangle(x0 + round(s1 \* mx), y0 - round(fun(u) \* my), x0 + round(s1 \* mx + l), y0);
73. **end**;
74. **end**;
75. **if** (x0 + round(s1 \* mx + l)) = (x0 + round(x \* mx)) **then**
76. **begin**
77. **if** y0 - round(fun(x) \* my) < y0 **then**
78. **begin**
79. setpencolor(clgreen);
80. **for var** v := x0 + round(s1 \* mx - l) **to** x0 + round(s2 \* mx) **do**
81. **begin**
82. u := p / 2 + x;
83. **if** y0 - round(fun(u) \* my)<y0 **then**
84. Rectangle(x0 + round(s1 \* mx + l), y0 - round(fun(u) \* my), x0 + round(s1 \* mx + (k + l)), y0);
85. **end**;
86. **end**;
87. l := l + k;
88. w := w + 1;
89. **if** (w + 1) = n **then**
90. **break**;
91. **end**;
92. x := x + 0.001;
93. **end**;
94. x := a;
95. **while** x <= b **do**
96. **begin**
97. setpixel(x0 + round(x \* mx), y0 - round(fun(x) \* my), clred);
98. x := x + 0.001;
99. **end**;
100. **end**;
101. **function** graphdop(**var** s1, s2, h: real; ras: integer): integer;
102. **var**
103. con: integer; s: string;
104. **begin**
105. con := 0;
106. **repeat**
107. s := 'Масштаб по x: ' + m1;
108. textout(0, 10, s);
109. s := 'Масштаб по у: ' + m2;
110. textout(0, 30, s);
111. textout(0, 50, 'Изменить маштаб по x - 1');
112. textout(0, 70, 'Изменить маштаб по y - 2');
113. textout(0, 90, 'Продолжить - 0');
114. read(input);
115. **case** input **of**
116. 1:
117. **begin**
118. textout(0, 130, 'Введите маштаб от 10 до 50');
119. readln(m1);
120. graph(s1, s2, h, ras);
121. **end**;
122. 2:
123. **begin**
124. textout(0, 130, 'Введите маштаб от 10 до 50');
125. readln(m2);
126. graph(s1, s2, h, ras);
127. **end**;
128. 0: con := 1;
129. **end**;
130. **until** con = 1;
131. graph(s1, s2, h, ras);
132. **end**;
133. **function** mainfun: integer;
134. **var**
135. a, b, h, f, x, s, pog: real; ss: string;
136. **begin**
137. clearwindow;
138. textout(0, 0, 'Вычисление площади фигуры, ограниченной кривой 1\*x^3+(-1)\*x^2+3\*x+11 и осью Ох (в положительной части по оси Оу)');
139. textout(0, 15, 'Введите пределы интегрирования a и b и количество разбиений n:');
140. readln(a, b, n);
141. ss := a + ' ' + b + ' ' + n;
142. textout(415, 15, ss);
143. h := (b - a) / n;
144. x := a + (h / 2);
145. m1 := 20;
146. m2 := 20;
147. **for var** i := 0 **to** n **do**
148. **begin**
149. f := fun(x);
150. s := s + f;
151. x := x + h;
152. **end**;
153. m1 := 20;
154. m2 := 20;
155. s := s \* h;
156. s := Round(s \* 1000) / 1000;
157. ss := 'Площадь=' + s;
158. textout(0, 50, ss);
159. textout(0, 70, 'Вывести погрешность?');
160. textout(0, 110, 'Да - 1');
161. textout(0, 130, 'Нет - 0');
162. read(input);
163. **case** input **of**
164. 1:
165. **begin**
166. pog :=
167. abs((fun1(b) - fun1(a)) - s);
168. pog := Round(pog \* 1000) / 1000;
169. ss := 'Погрешность=' + pog;
170. textout(0, 180, ss);
171. textout(0, 200, 'Нажмите enter чтоб продолжить');
172. readln();
173. readln;
174. **end**;
175. 0:
176. **begin**
177. textout(0, 150, 'Нажмите enter чтоб продолжить');
178. readln();
179. readln;
180. **end**;
181. **end**;
182. graph(a, b, h, n);
183. graphdop(a, b, h, n);
184. textout(0, 10, 'Запустить программу заново?');
185. textout(0, 30, 'Да - 1');
186. textout(0, 50, 'Нет - 0');
187. read(input);
188. **case** input **of**
189. 1: mainfun;
190. 0: **exit**();
191. **end**;
192. mainfun := 0;
193. **end**;
194. **begin**
195. MaximizeWindow;
196. Writeln('Вычисление площади фигуры, ограниченной кривой 1\*x^3+(-1)\*x^2+3\*x+11 и осью Ох (в положительной части по оси Оу)');
197. Writeln('Для ввода пределов интегрирования введите 1, для выхода введите 0');
198. input := vib(input);
199. **case** input **of**
200. 1: mainfun;
201. **end**;
202. **end**.
203. **Результат программы:**

****

**Рис. 1 – Результат программы**

1. **Вывод:** при выполнении данной домашней контрольной работы №4 мы узнали много нового и позновательного. Мы познакомились с модулем GraphABC и с его основными примитивами. Нам стало понятно, как представить функцию в виде графика, так же мы узнали, как выделить на графике необходимую область с помощью штриховки. Мы научились делать масштабирование графика, поняли, что необходимо для того, чтобы были подписи на осях, узнали, как сделать вывод информации о задании. Со всеми поставленными задачами мы справились успешно и делали их с особым энтузиазмом, потому что мы понимали, что это понадобится нам в будущем. Когда наша программа заработала мы были очень счастливы, как маленькие дети, а глаза наши сияли, как самый яркий алмаз.