|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  Калужский филиал  федерального государственного бюджетного  образовательного учреждения высшего образования  ***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»***  ***(КФ МГТУ им. Н.Э. Баумана)*** |

|  |  |
| --- | --- |
| **ФАКУЛЬТЕТ** | **ИУК «Информатика и управление»** |
| **КАФЕДРА** | **ИУК4 «Программное обеспечение ЭВМ,** |
| **информационные технологии»** | |

**ЛАБОРАТОРНАЯ РАБОТА №8**

**«Программирование графики»**

**ДИСЦИПЛИНА: «Системное программирование»**



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил: студент гр. ИУК4-31Б | |  |  | ( | Суриков Н. С. | ) |
|  |  |  | (подпись) |  | (Ф.И.О.) |  |
| Проверил: | |  |  | ( | Амеличева К. А. | ) |
|  |  |  | (подпись) |  | (Ф.И.О.) |  |

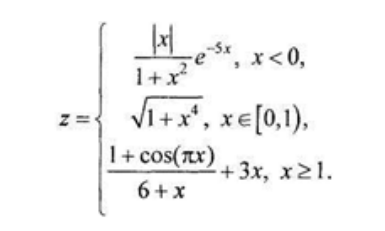
|  |  |
| --- | --- |
| Дата сдачи (защиты):  Результаты сдачи (защиты): | |
|  | - Балльная оценка:  - Оценка: |

**Цель:** практическое овладение навыками разработки программного кода на языке Ассемблер. Изучение основных принципов представление данных на мониторе компьютера в графическом виде.

**Задачи:**

1. Разработка программы, с использованием графических примитивов.

**Вариант №20**



**Листинг программы:**

1 .486

2 model use16 small

3 .stack 100h

4 .data

5 const\_neg5 dw -5 *; Константа -5*

6 const\_120 dw 213 *; Константа 120*

7 const\_240 dw 213 *; Константа 240*

8 const\_6 dw 6 *; Константа 6*

9 const\_3 dw 3 *; Константа 3*

10

11 screen\_middle dw 330 *; Центр экрана по оси Y*

12 var\_x dw ? *; Переменная для X*

13 var\_y dw ? *; Переменная для Y*

14 axis\_marker dw ? *; Метка текущей позиции оси*

15 scale\_y\_coeff dw 50 *; Коэффициент масштабирования по оси Y*

16

17 music2 dw 2100, 2000, 3000, 2000, 2100, 2000, 2500, 1500, 2500

18 dw 2000, 12000, 7000, 8000, 4000, 2217, 2637, 2349, 2093

19 dw 3729, 2959, 3729, 2959, 2793, 2637, 2349, 3951, 3322

20 dw 3136, 3322, 3136, 2793, 2349, 2637, 2217

21

22 music1 dw 2620, 2940, 3300, 3490, 3920, 4400, 4940, 5230

23 dw 5870, 6590, 6980, 7840, 8800, 9880, 10470, 11750

24 dw 13100, 13970, 15680, 17600, 19750, 20930, 22170, 23490

25 dw 24890, 26370, 27930, 29590, 31360, 33140, 34920, 36700

26 dw 38600

27

28 music\_pointer dw ?

29

30 delay\_count dw 1

31 melody\_end dw 30

32 melody\_max dw 20 *; Указатель на текущую мелодию*

33 .code

34 Start:

35 *; Инициализация сегментов данных*

36 mov ax, @data

37 mov ds, ax

38

39 *; Установим указатель на первую мелодию*

40 lea ax, music1

41 mov music\_pointer, ax

42

43 *; Переключение в графический режим 320x200*

44 xor ax, ax

45 mov al, 10h

46 int 10h

47

48 *; Заливка экрана белым цветом*

49 mov ax, 0600h *; Функция прокрутки вверх*

50 mov bh, 15 *; Белый цвет*

51 mov cx, 0000h *; Верхний левый угол*

52 mov dx, 184Fh *; Нижний правый угол*

53 int 10h

54

55 *; Рисуем вертикальную линию в центре экрана*

56 mov ah, 0Ch *; Установка графической точки*

57 mov al, 10 *; Зелёный цвет*

58 mov bh, 0h *; Номер видеостраницы*

59 mov cx, 400 *; Количество итераций по Y*

60 draw\_vertical\_line:

61 push cx

62 mov axis\_marker, cx *; Сохраняем текущую позицию оси*

63 mov dx, axis\_marker

64 mov cx, 213 *; Сдвиг оси вправо*

65 int 10h

66 pop cx

67 loop draw\_vertical\_line

68

69 *; Рисуем горизонтальную линию в центре экрана*

70 mov ah, 0Ch *; Установка графической точки*

71 mov al, 4 *; Красный цвет*

72 mov cx, 639 *; Количество итераций по X*

73 mov bh, 0h *; Номер видеостраницы*

74 mov dx, screen\_middle

75 draw\_horizontal\_line:

76 int 10h

77 loop draw\_horizontal\_line

78

79 *; стрелка X*

80 mov al, 4 *; Цвет стрелки (красный)*

81 mov cx, 639

82 mov dx, 330

83 int 10h

84

85 mov dx, 331

86 int 10h

87

88 mov dx, 332

89 int 10h

90

91 mov cx, 638

92 mov dx, 329

93 int 10h

94

95 mov dx, 333

96 int 10h

97

98 mov cx, 637

99 mov dx, 328

100 int 10h

101

102 mov dx, 335

103 int 10h

104 *; стрелка Y*

105 mov al, 10 *; Цвет стрелки (зелёный)*

106 mov cx, 213

107 mov dx, 1

108 int 10h

109

110 mov cx, 214

111 int 10h

112

113 mov cx, 215

114 int 10h

115

116 mov cx, 212

117 mov dx, 2

118 int 10h

119

120 mov cx, 216

121 int 10h

122

123 mov cx, 211

124 mov dx, 3

125 int 10h

126

127 mov cx, 217

128 int 10h

129 *; Начинаем расчёт и отрисовку графика*

130 mov cx, 639 *; Устанавливаем начальное значение X*

131 xor di, di

132 draw\_graph:

133 mov var\_x, cx

134 fild var\_x *; Загружаем X в стек FPU*

135 fild const\_120 *; Загружаем 120 в стек FPU*

136 fsub *; Вычитаем (X - 120)*

137 fild const\_240 *; Загружаем 240 в стек FPU*

138 fdiv *; Делим на 240 (X - 120) / 240*

139

140 *; Условие: если X < 0*

141 fld st(0)

142 fldz *; Загружаем 0*

143 fcom *; Сравниваем X с 0*

144 fstsw ax *; Сохраняем результат сравнения*

145 sahf *; Загружаем флаги*

146 ja less\_than\_zero *; Если X < 0, перейти на обработку*

147

148 *; Условие: 0 <= X < 1*

149 fld st(1)

150 fld1

151 fsub

152 fldz *; Загружаем 0*

153 fcom *; Сравниваем X с 0*

154 fstsw ax *; Сохраняем результат сравнения*

155 sahf *; Загружаем флаги*

156 ja between\_zero\_and\_one *; Если X < 0, перейти на обработку*

157

158 *; Если X >= 1*

159 jmp greater\_than\_one

160

161 less\_than\_zero:

162 fld st(1) *; Загружаем x из стека.*

163 fabs *; Берём модуль ST(0) (|x|).*

164

165 fld st(2) *; Загружаем (x - 213) / 213.*

166 fld st(0) *; Дублируем вершину стека.*

167 fmul *; Умножаем (x - 213) / 213 на (x - 213) / 213.*

168

169 fld1 *; Загружаем константу 1.*

170 fadd *; Прибавляем 1 к произведению.*

171 fdiv *; Делим модифицированный результат на (x - 213) / 213.*

172

173 fld st(2) *; Берём (x - 213) / 213.*

174 fild const\_neg5 *; Загружаем константу -5.*

175 fmul *; Умножаем на -5.*

176

177 fldl2e *; Загружаем log2(e).*

178 fmul *; Умножаем результат на log2(e).*

179 fld st *; Дублируем вершину стека.*

180

181 frndint *; Округляем значение.*

182 fsub st(1), st *; Вычитаем округлённое значение.*

183 fxch st(1) *; Меняем местами ST(0) и ST(1).*

184

185 f2xm1 *; Вычисляем 2^(ST(0)) - 1.*

186 fld1 *; Загружаем 1.*

187 faddp st(1), st *; Прибавляем 1 к 2^(ST(0)) - 1.*

188 fscale *; Умножаем на 2^(целая часть ST(1)).*

189

190 fstp st(1) *; Сохраняем результат.*

191 fmul *; Умножаем текущий результат на предыдущий.*

192

193 mov al, 5

194

195 jmp calculate

196 *; Обработка при 0 <= X <= 1*

197 between\_zero\_and\_one:

198 fld st(3)

199 fld st(0)

200 fmul *; X^2*

201 fld st(0)

202 fmul

203 fld1

204 fadd *; X^2 + 1*

205 fsqrt *; sqrt(X^2 + 1)*

206 mov al, 3

207

208 jmp calculate

209

210 greater\_than\_one:

211 fld st(3)

212 fld st(0)

213 fldpi *; Загружаем Pi*

214 fmul *; X \* Pi*

215 fcos *; cos(X \* Pi)*

216 fld1

217 fadd *; cos(X \* Pi) + 1*

218 fld st(1)

219 fild const\_6

220 fadd *; cos(X \* Pi) + 1 + 6*

221 fdiv *; Нормализация*

222 fxch st(1)

223 fild const\_3

224 fmul *; Умножение на 3*

225 fadd *; Результат*

226

227 mov al, 10

228 jmp calculate

229

230 calculate:

231 *; Преобразуем координаты для отображения на экране*

232 fimul scale\_y\_coeff *; Умножаем на коэффициент Y*

233 fchs *; Изменяем знак*

234 fiadd screen\_middle *; Смещаем относительно центра экрана*

235 frndint *; Округляем до целого*

236 fistp var\_y *; Сохраняем результат в var\_y*

237

238 *; Рисуем точку графика*

239 push cx

240 mov cx, var\_x

241 mov ah, 0Ch *; Установка графической точки*

242 mov bh, 0h *; Номер видеостраницы*

243 mov dx, var\_y

244 *; mov al, 0 ; Чёрный цвет*

245 int 10h

246 pop cx

247

248 *; Задержка*

249 push cx

250 mov cx, delay\_count *; Используем переменную для настраиваемой задержки*

251 delay:

252 push cx

253 xor cx,cx

254 delay\_loop:

255 nop *; Минимальная инструкция для задержки*

256 loop delay\_loop

257 pop cx

258 loop delay

259 pop cx

260

261 *; Настройка музыкального таймера*

262 mov al, 10110110b *; Управляющее слово: канал 2, режим 3*

263 out 43h, al

264 push di

265 push si

266 mov di, music\_pointer *; Загружаем указатель на текущую мелодию*

267 add si, di

268 mov ax, [si] *; Загружаем данные мелодии*

269 pop si

270 pop di

271 add di, 2 *; Смещаем указатель мелодии*

272 cmp di, melody\_end *; Проверяем конец мелодии*

273 jge reset\_di\_pointer

274 jmp process\_music

275 reset\_di\_pointer:

276 xor di, di *; Сбрасываем указатель*

277 add si, 2

278 cmp si, melody\_max

279 je reset\_si\_pointer

280 jmp process\_music

281 reset\_si\_pointer:

282 mov si, 0 *; Сбрасываем второй указатель*

283 process\_music:

284 out 42h, al *; Отправляем младший байт частоты*

285 mov al, ah *; Отправляем старший байт частоты*

286 out 42h, al

287

288 *; Включаем динамик*

289 in al, 61h *; Читаем порт управления*

290 or al, 00000011b *; Устанавливаем биты включения канала 2*

291 out 61h, al

292

293 *; Проверка ввода с клавиатуры*

294 mov ah, 01h *; Проверяем, нажата ли клавиша*

295 int 16h

296 jz skip\_key\_check *; Если клавиша не нажата, пропустить проверку*

297

298 mov ah, 00h *; Чтение нажатой клавиши*

299 int 16h

300 cmp al, '0' *; Если нажата '0', выйти*

301 je exit\_program

302 cmp al, '1' *; Если нажата '1', сменить мелодию на первую*

303 je set\_music1

304 cmp al, '2' *; Если нажата '2', сменить мелодию на вторую*

305 je set\_music2

306

307 set\_music1:

308 lea ax, music1

309 mov music\_pointer, ax

310 jmp skip\_key\_check

311 set\_music2:

312 lea ax, music2

313 mov music\_pointer, ax

314

315 skip\_key\_check:

316 dec cx

317 jnz draw\_graph

318

319 exit\_program:

320 *; Завершаем работу программы*

321 in al, 61h

322 and al, 0FCh *; Выключаем динамик*

323 out 61h, al

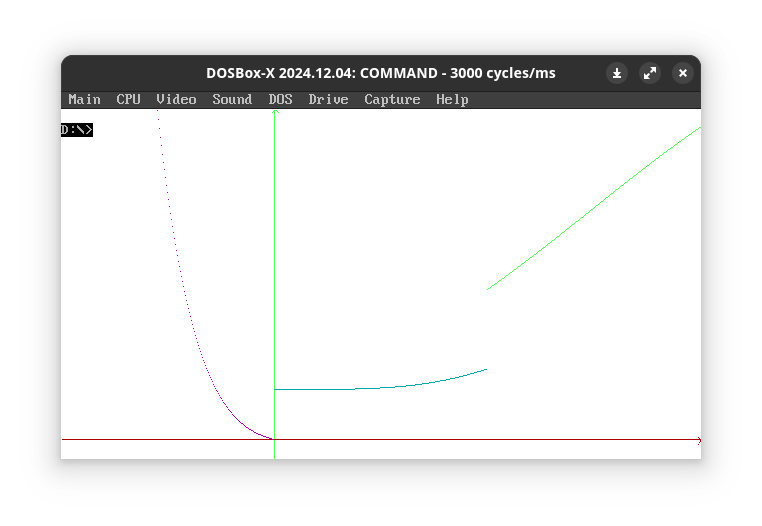
324

325 mov ah, 4Ch *; Завершение программы*

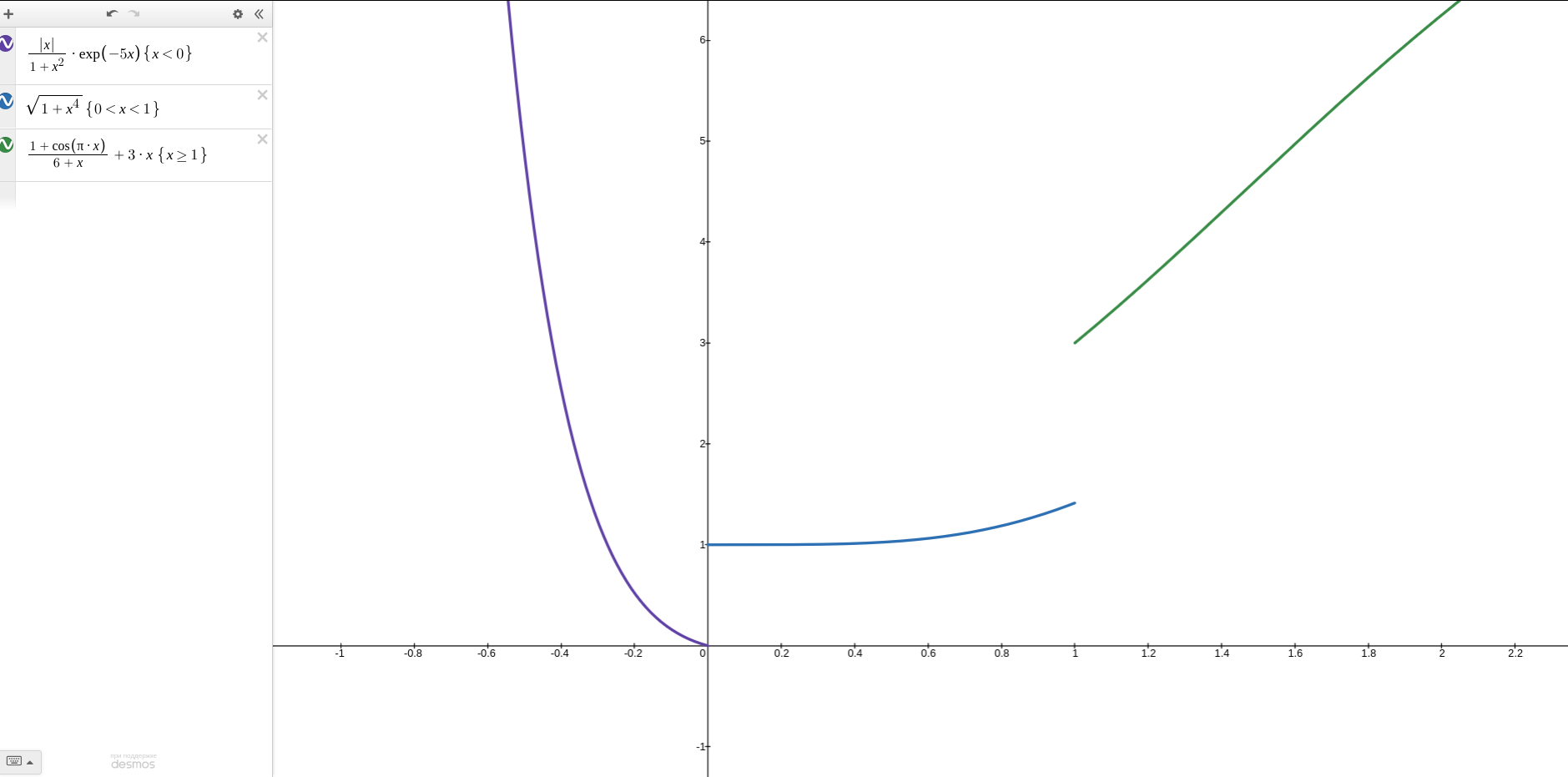
326 int 21h

327 end Start

**Результат выполнения программы:**

****

**Desmos:**

****

**Вывод:** в ходе выполнения работы были сформированы практические навыки разработки программного кода на языке Ассемблера; изучены основные принципы представления данных на мониторе компьютера в графическом виде.