```
import turtle

##全局变量##
#词频排列显示个数
count = 10
#单词频率数组-作为y轴数据
data = []
#单词数组-作为x轴数据
words = []
#y轴显示放大倍数-可以根据词频数量进行调节
yScale = 6
#x轴显示放大倍数-可以根据count数量进行调节
xScale = 30

################# Turtle Start  ####################
#从点(x1,y1)到(x2,y2)绘制线段
def drawLine(t, x1, y1, x2, y2):
    t.penup()
    t.goto (x1, y1)
    t.pendown()
    t.goto (x2, y2)

# 在坐标(x,y)处写文字
def drawText(t, x, y, text):
    t.penup()
    t.goto (x, y)
    t.pendown()
    t.write(text)

def drawGraph(t):
    #绘制x/y轴线
    drawLine (t, 0, 0, 360, 0)
    drawLine (t, 0, 300, 0, 0)

    #x轴: 坐标及描述
    for x in range(count):
        x=x+1 #向右移一位,为了不画在原点上
        drawText(t, x*xScale-4, -20, (words[x-1]))
        drawText(t, x*xScale-4, data[x-1]*yScale+10, data[x-1])
    drawBar(t)

#绘制一个柱体
def drawRectangle(t, x, y):
    x = x*xScale
    y = y*yScale#放大倍数显示
    drawLine(t, x-5, 0, x-5, y)
    drawLine(t, x-5, y, x+5, y)
    drawLine(t, x+5, y, x+5, 0)
    drawLine(t, x+5, 0, x-5, 0)

#绘制多个柱体
def drawBar(t):
    for i in range(count):
        drawRectangle(t, i+1, data[i])
################# Turtle End  ####################


#对文本的每一行计算词频的函数
def processLine(line, wordCounts):
    #用空格替换标点符号
    line = replacePunctuations(line)
    #从每一行获取每个词
    words = line.split()
    for word in words:
        if word in wordCounts:
            wordCounts[word] += 1
        else:
            wordCounts[word] = 1

#空格替换标点的函数
def replacePunctuations(line):
    for ch in line:
        if ch in "~@#$%^&*()_-+=<>?/,.:;{}[]|\'""":
            line = line.replace(ch, " ")
    return line

def main():
    #用户输入一个文件名
    filename = input("enter a filename:").strip()
    infile = open(filename, "r")

    #建立用于计算词频的空字典
    wordCounts = {}
    for line in infile:
        processLine(line.lower(), wordCounts)

    #从字典中获取数据对
    pairs = list(wordCounts.items())

    #列表中的数据对交换位置,数据对排序
    items = [[x,y]for (y,x)in pairs]
    items.sort()

    #输出count个数词频结果
    for i in range(len(items)-1, len(items)-count-1, -1):
        print(items[i][1]+"\t"+str(items[i][0]))
        data.append(items[i][0])
        words.append(items[i][1])

    infile.close()

    #根据词频结果绘制柱状图
    turtle.title('词频结果柱状图')
    turtle.setup(900, 750, 0, 0)
    t = turtle.Turtle()
    t.hideturtle()
    t.width(3)
    drawGraph(t)

#调用main()函数
if __name__ == '__main__':
    main()
```