1. Сначала создадим структуру файла font_ready.json

Ваш JSON файл должен содержать массив чисел, представляющих шрифт. Вот как он может выглядеть:

```
[
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x3e, 0x45, 0x51, 0x45, 0x3e,
  0x00, 0x3e, 0x6b, 0x6f, 0x6b, 0x3e
]
```

2. Программа для чтения и визуализации шрифта

Вот Python-скрипт, который считывает этот файл и создает изображение:

```
import json
from PIL import Image, ImageDraw

def draw_font_from_json(json_file, output_image):
  # Чтение данных из JSON файла
  with open(json_file, 'r') as f:
    font_data = json.load(f)

  # Размер одного символа
  char_width = 6
  char_height = 8

  # Количество символов в данных
  num_chars = len(font_data) // (char_width * char_height // 8)
  # Поскольку у вас 6 строк на символ (видимо, 6 байт на символ для 6x8)
  # Уточнение: в вашем примере 6 строк по 1 байту (8 бит), но используется только 6 бит?
  # Будем считать, что каждый символ представлен 6 байтами (для 6x8)
  num_chars = len(font_data) // 6

  # Создаем изображение
  # Размер изображения: (ширина символов + промежутки) * количество символов
  img_width = num_chars * (char_width + 1)
  img_height = char_height
```

```
    img = Image.new('1', (img_width, img_height), color=0)  # '1' для бинарного изображения
    draw = ImageDraw.Draw(img)

    # Рисуем каждый символ
    for char_idx in range(num_chars):
        # Получаем 6 байт для текущего символа
        char_bytes = font_data[char_idx*6 : (char_idx+1)*6]

        # Рисуем каждый пиксель
        for y in range(6):  # 6 строк
            byte = char_bytes[y]
            for x in range(6):  # 6 бит (используем только младшие 6 бит)
                if byte & (1 << (7 - x)):  # Проверяем бит (уточнение может потребоваться)
                    # Рисуем пиксель
                    px_x = char_idx * (char_width + 1) + x
                    px_y = y  # +1 если нужно смещение
                    if px_x < img_width and px_y < img_height:
                        img.putpixel((px_x, px_y), 1)

    # Сохраняем изображение
    img.save(output_image)
    print(f"Изображение сохранено как {output_image}")

# Использование функции
draw_font_from_json('font_ready.json', 'font_image.png')
```

3. Объяснение работы программы:

1. **Чтение данных**: Программа считывает массив чисел из JSON-файла.
2. **Параметры шрифта**: Размер символа 6x8 пикселей, каждый символ представлен 6 байтами (по одному на каждую строку).
3. **Создание изображения**: Создается бинарное изображение (1 бит на пиксель), где белый пиксель - 1, черный - 0.
4. **Отрисовка символов**: Для каждого символа проверяются биты в каждом байте, и ставятся соответствующие пиксели.

5. **Сохранение**: Результат сохраняется в PNG-файл.

4. Замечания:

1. В вашем примере данные содержат 18 значений, что соответствует 3 символам (18 / 6 = 3).
2. Я предположил, что каждый байт представляет вертикальную линию из 8 бит, но используется только 6 из них (для высоты 6). Возможно, потребуется уточнение этого момента.
3. Если порядок бит или байт нужно изменить, вам может потребоваться настроить логику проверки битов.

5. Альтернативный вариант (если данные организованы иначе):

Если каждый символ представлен 6 значениями, где каждое значение - это строка из 6 пикселей (а не 8), то код нужно изменить:

```
# В этом случае изменяем часть кода, которая рисует пиксели:

for y in range(6):  # 6 строк

   row = char_bytes[y]

   for x in range(6):  # 6 пикселей в строке

      if row & (1 << x):  # Проверяем бит (зависит от порядка)

         px_x = char_idx * (char_width + 1) + x

         px_y = y

         img.putpixel((px_x, px_y), 1)
```

Попробуйте оба варианта, чтобы увидеть, какой дает правильный результат.