Explicação do Código Passo a Passo

1. Definição da Estrutura do Nó

```
typedef struct No {
    char texto[100];       // Pergunta ou decisão que o
nó representa
    struct No* sim;        // Ponteiro para o nó que
responde "sim"
    struct No* nao;        // Ponteiro para o nó que
responde "não"
} No;
```

Cada nó da árvore contém um texto que pode ser uma pergunta (ex: "Clima: Ensolarado?") ou uma decisão final (ex: "Jogar tênis (Manhã - Ensolarado)"). Além disso, cada nó aponta para dois nós filhos: um para a resposta "sim" e outro para a resposta "não".

---

2. Função para Criar um Nó

```
No* criarNo(const char* texto) {
    No* novo = (No*) malloc(sizeof(No));
    strcpy(novo->texto, texto);
    novo->sim = NULL;
    novo->nao = NULL;
    return novo;
}
```

Esta função cria um novo nó, alocando memória, copiando o texto passado e iniciando os ponteiros `sim` e `nao` como `NULL`.

---

3. Função para Perguntar ao Usuário e Receber Resposta "sim" ou "não"

```
int perguntaSimNao(const char* texto) {
    char resposta[10];
    while (1) {
        printf("%s (sim/nao): ", texto);
        scanf("%9s", resposta);
        if (strcmp(resposta, "sim") == 0) return 1;
        else if (strcmp(resposta, "nao") == 0) return 0;
        else printf("Resposta inválida! Digite 'sim' ou
'nao'.\n");
    }
}
```

Esta função exibe uma pergunta e espera o usuário digitar "sim" ou "nao".
Só retorna quando a resposta for válida.

---

4. Função para Percorrer a Árvore e Tomar Decisões

```
void percorrerArvore(No* no) {
    if (!no) return;

    // Se não tem filhos, é uma decisão final
    if (no->sim == NULL && no->nao == NULL) {
        printf("Decisão: %s\n", no->texto);
        return;
    }

    // Se tem filhos, faz a pergunta do nó
    int resposta = perguntaSimNao(no->texto);
    if (resposta == 1) {
        percorrerArvore(no->sim);
    } else {
        percorrerArvore(no->nao);
```

```
    }
}
```

Esta função é recursiva e vai descendo na árvore conforme a resposta do usuário.
Quando chega em um nó folha (sem filhos), imprime a decisão.

---

5. Função que Cria a Árvore com as Perguntas e Decisões

Essa parte monta toda a árvore, criando nós para perguntas (ex: "Clima: Ensolarado?") e para decisões finais (ex: "Jogar tênis (Manhã - Ensolarado)").
Também conecta os nós entre si.

Exemplo:

```
No* jogarManhaEnsolarado = criarNo("Jogar tênis (Manhã - Ensolarado)");
No* climaManhaEnsolarado = criarNo("Clima: Ensolarado?");
climaManhaEnsolarado->sim = jogarManhaEnsolarado;
```

Aqui, se a resposta para "Clima: Ensolarado?" for "sim", o programa vai para o nó que indica jogar tênis pela manhã com clima ensolarado.

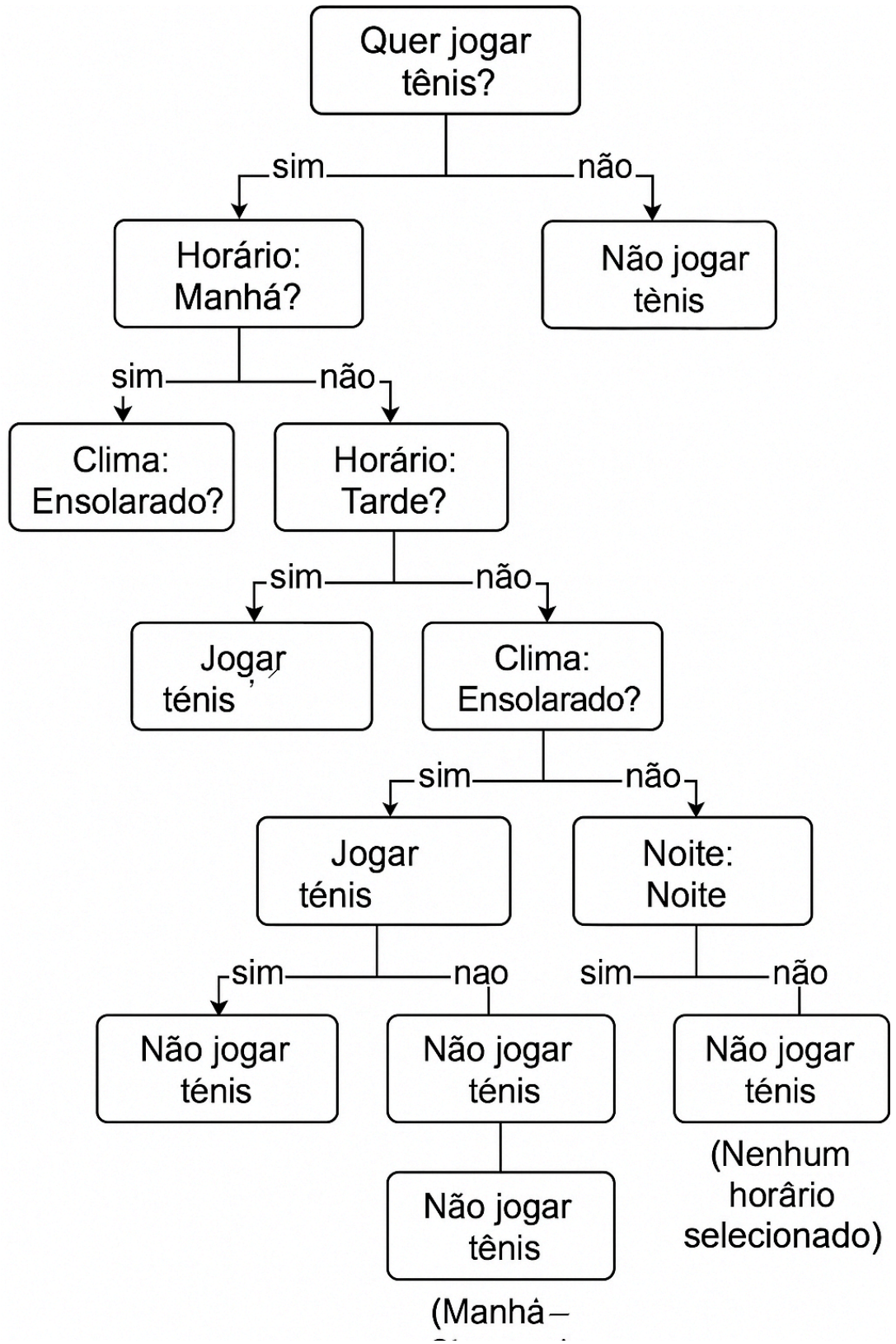
---

6. Função Principal (`main`) que Executa a Árvore de Decisão

```
int main() {
    No* raiz = criarArvoreDecisao();
    printf("Árvore de decisão para jogar tênis:\n");
    percorrerArvore(raiz);
    return 0;
}
```

Aqui, a árvore é criada e começa a interação com o usuário para decidir se ele vai jogar tênis ou não, baseado nas respostas.

Desenho da Árvore de Decisão (Texto)



- Cada nó representa uma pergunta ou decisão.
- Os caminhos "sim" ou "não" levam a nós diferentes conforme a resposta.
- Ao chegar a um nó sem filhos, uma decisão final é mostrada.