Questão 1

A)

Os arquivos de texto são escritos em ASCII 2 fazendo com que sejam compativeis com editores de texto, porém sejam maiores em tamanho já que caracteres ocupam mais espaço. Usam as funções fprintf, fscanf, fgets, fputs. Os binários são escritos com dados na forma bruta, ocupam menos espaço e são mais rápidos de gravar e ler. Mais adequados para estruturas de dados complexas e grandes volumes de dados. Usam fwrite, fread.

B)

Os arquivos permitem que dados sejam persistentes ou seja, sejam lembrados entre sessões.

Questão 2

A leitura sequencial percorre o arquivo do inicio ao fim, passo a passo. É mais lenta já que é necessário percorrer todos os dados até que se chegue no objetivo.

Exemplo:

```
#include <stdio.h>

int main() {
    FILE *arq = fopen("dados.txt", "r");
    char linha[100];

    while (fgets(linha, sizeof(linha), arq)) {
        printf("%s", linha);
    }

    fclose(arq);
    return 0;
}
```

A leitura randômica permite que o programa pule partes dos dados e vá direto à posição desejada. Isso pode ser feito usando o fseek().

```
#include <stdio.h>

int main() {
    FILE *arq = fopen("dados.txt", "r");
    char buffer[20];

    fseek(arq, 10, SEEK_SET); // Vai para o 10° byte do arquivo
    fread(buffer, sizeof(char), 10, arq);
    buffer[10] = '\0';
```

```
    printf("Leitura randômica: %s\n", buffer);

    fclose(arq);
    return 0;
}
```

Questão 3

Modos de abertura de arquivos em C (fopen)

1. **“r”**
    - Abre um arquivo para **leitura**.
    - O arquivo **deve existir**, caso contrário fopen retorna NULL.
2. **“w”**
    - Abre um arquivo para **escrita**.
    - Se o arquivo já existir, seu conteúdo é **apagado**.
    - Se não existir, um novo arquivo é criado.
3. **“a”**
    - Abre um arquivo para **acréscimo (append)**.
    - Os dados sempre serão gravados **no final do arquivo**.
    - Se não existir, é criado.
4. **“r+”**
    - Abre um arquivo para **leitura e escrita**.
    - O arquivo deve existir.
    - Permite alterar dados já existentes sem apagar o conteúdo.
5. **“w+”**
    - Abre um arquivo para **leitura e escrita**.
    - Se o arquivo já existir, seu conteúdo é **apagado**.
    - Se não existir, é criado.

4.c

```c
#include <stdio.h>

int main() {
  FILE *saida;

  saida = fopen("saida.txt", "w");

  fputs("Primeiro arquivo em C", saida);

  fclose(saida);

  return 0;
}
```

5.c

```c
#include <stdio.h>

int main() {

  FILE *dados = fopen("dados.txt", "r");
  char conteudo[100];

  if (dados == NULL) {
    printf("Erro ao abrir (arquivo não existe)");
    return 1;
  }

  fgets(conteudo, sizeof(conteudo), dados);

  printf("%s", conteudo);

  fclose(dados);

  return 0;
}
```

6.c

```c
#include <stdio.h>

int main() {
  FILE *frase = fopen("frase.txt", "w");
  if (frase == NULL) {
      printf("Erro ao abrir o arquivo!\n");
      return 1;
  }
  char buffer[100];

  printf("Digite uma frase: ");
  fgets(buffer, sizeof(buffer), stdin);

  for (int i = 0; buffer[i] != '\0'; i++) {
      fputc(buffer[i], frase);
  }

  fclose(frase);

  return 0;
}
```

7.c

```c
#include <stdio.h>

int main() {
    FILE *frase = fopen("frase.txt", "r");
```

```
    int c;

    while ((c = fgetc(frase)) != EOF)
        printf("%c", c);

    fclose(frase);

    return 0;
}
```

8.c

```
#include <stdio.h>

int main() {
    FILE *texto = fopen("texto.txt", "r");
    int quantidade = 0;
    int c;

    if (texto == NULL) {
        printf("Erro ao abrir o arquivo \n");
        return 1;
    }

    while ((c = fgetc(texto)) != EOF) {
        quantidade++;
    }

    fclose(texto);
    printf("%d", quantidade);
    return 0;
}
```

9.c

```
#include <stdio.h>
#include <string.h>

int main() {
  FILE *frase = fopen("frase.txt", "w");
  if (frase == NULL) {
      printf("Erro ao abrir o arquivo!\n");
      return 1;
  }
  char buffer[100];


  printf("Digite uma frase: ");
  fgets(buffer, sizeof(buffer), stdin);
```

```c
  while (strcmp(buffer, "fim\n") != 0) {
      fputs(buffer, frase);
      printf("Digite uma frase: ");
      fgets(buffer, sizeof(buffer), stdin);
  }


  fclose(frase);

  return 0;
}
```

10.c

```c
#include <stdio.h>

int main() {
    FILE *frase = fopen("frase.txt", "r");
    char buffer[256];

    if (frase == NULL) {
        printf("Erro ao abrir o arquivo!\n");
        return 1;
    }


    while (fgets(buffer, sizeof(buffer), frase) != NULL) {
        printf("%s", buffer);
    }


    fclose(frase);
    return 0;
}
```

11.c

```c
#include <stdio.h>

int main() {
    FILE *arquivo = fopen("numeros.txt", "w");
    int numero;

    for (int i = 0; i < 10; i++) {
        puts("Digite um número:");
        scanf("%d", &numero);
        fprintf(arquivo, "%d\n", numero);
    }

    fclose(arquivo);

    arquivo = fopen("numeros.txt", "r");
```

```c
    while (fscanf(arquivo, "%d", &numero) == 1) {
        printf("Número lido: %d\n", numero);
    }

    fclose(arquivo);
    return 0;
}
```

12.c

```c
#include <stdio.h>

int main() {
    FILE *dados = fopen("dados.bin", "wb");
    float valor;

    if (dados == NULL) {
        printf("Erro ao abrir o arquivo.\n");
        return 1;
    }


    for (int i = 0; i < 10; i++) {
        printf("Digite o valor %d: ", i + 1);
        scanf("%f", &valor);
        fwrite(&valor, sizeof(float), 1, dados);
    }

    fclose(dados);

    FILE *leitura = fopen("dados.bin", "rb");

    if (leitura == NULL) {
        printf("Erro ao abrir o arquivo para leitura.\n");
        return 1;
    }

    float valor_lido;
    while(fread(&valor_lido, sizeof(float), 1, leitura) == 1) {
        printf("Valor lido: %.2f\n", valor_lido);
    };

    fclose(leitura);

    return 0;
}
```

13.c

```c
#include <stdio.h>

typedef struct aluno {
```

```c
    char nome[50];
    char matricula[10];
    float media;
} Aluno;

int main() {
    FILE *alunos = fopen("alunos.bin", "wb");
    Aluno aluno;

    for (int i = 0; i < 3; i++) {
        printf("Digite o nome do aluno %d: ", i + 1);
        scanf("%s", aluno.nome);
        printf("Digite a matricula do aluno %d: ", i + 1);
        scanf("%s", aluno.matricula);
        printf("Digite a media do aluno %d: ", i + 1);
        scanf("%f", &aluno.media);
        fwrite(&aluno, sizeof(Aluno), 1, alunos);
    }

    fclose(alunos);

    FILE *alunos2 = fopen("alunos.bin", "rb");

    while (fread(&aluno, sizeof(Aluno), 1, alunos2) == 1) {
        printf("\nNome: %s\n", aluno.nome);
        printf("Matricula: %s\n", aluno.matricula);
        printf("Media: %.2f\n", aluno.media);
    }

    fclose(alunos2);

    return 0;
}
```

14.c

```c
#include <stdio.h>

typedef struct {
    char nome[50];
    char idade[4];
    float salario;
} Pessoa;

int main() {
    FILE *pessoas = fopen("pessoas.txt", "w");
    if (pessoas == NULL) {
        printf("Erro ao abrir o arquivo.\n");
        return 1;
    }
```

```
    Pessoa p;

    for(int i = 0; i < 5; i++) {
        printf("Digite o nome da pessoa %d: ", i+1);
        scanf("%s", p.nome);
        printf("Digite a idade da pessoa %d: ", i+1);
        scanf("%s", p.idade);
        printf("Digite o salario da pessoa %d: ", i+1);
        scanf("%f", &p.salario);

        fprintf(pessoas, "%s %s %.2f\n", p.nome, p.idade, p.salario);
    }


    fclose(pessoas);
    return 0;
}
```

15.c

```
#include <stdio.h>

int main() {
    FILE *pessoas = fopen("pessoas.txt", "r");
    char nome[100], sobrenome[100];
    float idade;

    while(fscanf(pessoas, "%s %s %f", nome, sobrenome, &idade) == 3) {
        printf("nome: %s, idade: %s, salario: %.2f\n", nome,
sobrenome, idade);
    }

    return 0;
}
```

16.c

```
#include <stdio.h>

int main() {
    char quinto, decimo, ultimo;
    FILE *texto = fopen("texto.txt", "r");
    if (texto == NULL) return 1;

    fseek(texto, 4, SEEK_SET);
    fread(&quinto, sizeof(char), 1, texto);
    fseek(texto, 9, SEEK_SET);
    fread(&decimo, sizeof(char), 1, texto);
    fseek(texto, -1, SEEK_END);
    fread(&ultimo, sizeof(char), 1, texto);
```

```c
    printf("Quinto caractere: %c\n", quinto);
    printf("Decimo caractere: %c\n", decimo);
    printf("Ultimo caractere: %c\n", ultimo);

    return 0;
}
```

17.c

```c
#include <stdio.h>

int main() {
    FILE *texto = fopen("texto.txt", "r");
    char buffer[11];

    fread(buffer, sizeof(char), 10, texto);

    printf("%s\n", buffer);
    rewind(texto);



    fread(buffer, sizeof(char), 10, texto);
    printf("%s\n", buffer);



    fclose(texto);


    return 0;
}
```

18.c

```c
#include <stdio.h>

int main() {
    FILE *origem = fopen("origem.txt", "r");
    FILE *copia = fopen("copia.txt", "w");
    int c;

    if (origem == NULL || copia == NULL) {
        printf("Erro ao abrir os arquivos.\n");
        return 1;
    }

    while ((c = fgetc(origem)) != EOF) {
        fputc(c, copia);
    }
```

```c
    fclose(origem);
    fclose(copia);


    puts("Copia realizada com sucesso!");

    return 0;
}
```

19.c

```c
#include <stdio.h>

int main() {
    FILE *origem = fopen("origem.txt", "r");
    FILE *copia = fopen("copia.txt", "r");
    FILE *final = fopen("final.txt", "w");
    int c;

    if (origem == NULL || copia == NULL || final == NULL) {
        printf("Erro ao abrir os arquivos.\n");
        return 1;
    }

    while ((c = fgetc(origem)) != EOF) {
        fputc(c, final);
    }

    while ((c = fgetc(copia)) != EOF) {
        fputc(c, final);
    }


    fclose(origem);
    fclose(copia);
    fclose(final);

    puts("concatenação realizada com sucesso!");

    return 0;
}
```

20.c

```c
#include <stdio.h>

int main() {

    if (remove("final.txt") == 0) {
        puts("Arquivo removido com sucesso");
    } else {
        puts("Erro ao remover arquivo");
```

```
    }

    return 0;
}
```

21.c

```
#include <stdio.h>

int main() {
    FILE *file = fopen("texto.txt", "r");
    if (file == NULL) {
        printf("Erro ao abrir arquivo.\n");
        return 1;
    }

    int quantidade = 0;
    int dentro_palavra = 0;
    char c;

    while ((c = fgetc(file)) != EOF) {
        if (c == ' ' || c == '\n' || c == '\t') {
            dentro_palavra = 0;
        } else if (!dentro_palavra) {
            dentro_palavra = 1;
            quantidade++;
        }
    }

    printf("%d\n", quantidade);

    fclose(file);
    return 0;
}
```

22.c

```
#include <stdio.h>

int main() {
    FILE *file = fopen("texto.txt", "r");
    if (file == NULL) {
        printf("Erro ao abrir arquivo.\n");
        return 1;
    }

    int quantidade = 0;
    char c;

    while ((c = fgetc(file)) != EOF) {
        if (c == '\n') {
            quantidade++;
```

```
        }
    }

    quantidade++;

    printf("%d", quantidade);

    fclose(file);

    return 0;
}
```

23.c

```
#include <stdio.h>

int main() {
    FILE *file = fopen("texto.txt", "r");
    if (file == NULL) {
        printf("Erro ao abrir arquivo.\n");
        return 1;
    }

    int contador = 0;
    int maior = 0;
    char c;

    while ((c = fgetc(file)) != EOF) {
        if (c != ' ' && c != '\n' && c != '\t') {
            contador++;
        } else {
            if (contador > maior) {
                maior = contador;
            }
            contador = 0;
        }
    }

    if (contador > maior)
        maior = contador;

    printf("%d\n", maior);

    fclose(file);
    return 0;
}
```

24.c

```
#include <stdio.h>

int main() {
```

```c
    FILE *file = fopen("texto.txt", "r");
    if (file == NULL) {
        printf("Erro ao abrir arquivo.\n");
        return 1;
    }

    int contagem[26] = {0};
    char c;

    while ((c = fgetc(file)) != EOF) {
       if (c >= 'A' && c <= 'Z') {
            c = c + ('a' - 'A');
       }

       if (c >= 'a' && c <= 'z') {
           contagem[c - 'a']++;
       }
    }


    fclose(file);

    for (int i = 0; i < 26; i++) {
        printf("%c: %d\n", 'a' + i, contagem[i]);
    }
    return 0;
}
```

25.c

```c
#include <string.h>
#include <stdio.h>

int main() {
    FILE *file = fopen("texto.txt", "r+");
    if (file == NULL) {
        printf("Erro ao abrir arquivo.\n");
        return 1;
    }

    char c;
    int i = 0;
    long pos;
    char palavra[100];

    while ((c = fgetc(file)) != EOF) {
        if (c >= 'A' && c <= 'Z') c = c + ('a' - 'A');

        if ((c >= 'a' && c <= 'z')) {
            if (i == 0)
                pos = ftell(file) - 1;
```

```
            palavra[i++] = c;
        } else {
            if (i > 0) {
                palavra[i] = '\0';
                if (strcmp(palavra, "casa") == 0) {
                    fseek(file, pos, SEEK_SET);
                    fputs("lar", file);
                    fputc(' ', file);
                }
                i = 0;
            }

            pos = ftell(file);
        }
    }


    fclose(file);
    return 0;
}
```

26.c

```
#include <stdio.h>
#include <string.h>

int main() {
    FILE *file = fopen("texto.txt", "r");
    if (file == NULL) {
        printf("Erro ao abrir o arquivo.\n");
        return 1;
    }

    char busca[100];
    char palavra[100];

    printf("Digite a palavra a ser buscada: ");
    fgets(busca, sizeof(busca), stdin);
    busca[strcspn(busca, "\n")] = '\0';

    for (int j = 0; busca[j]; j++) {
        if (busca[j] >= 'A' && busca[j] <= 'Z') {
            busca[j] += ('a' - 'A');
        }
    }

    char c;
    int i = 0;

    while ((c = fgetc(file)) != EOF) {
        if (c >= 'A' && c <= 'Z') c = c + ('a' - 'A');
```

```
        if (c ⩾ 'a' && c ⩽ 'z') {
            palavra[i++] = c;
        } else {
            palavra[i] = '\0';
            if (strstr(palavra, busca) ≠ NULL) {
                printf("Sua palavra foi encontrada na posição %ld\n",
ftell(file));
            }
            i = 0;
        }
    }

    if (i > 0) {
        palavra[i] = '\0';
        if (strstr(palavra, busca) ≠ NULL) {
            printf("%s\n", palavra);
        }
    }

    fclose(file);
    return 0;
}
```

27.c

```
#include <stdio.h>

int main() {
    FILE *file = fopen("texto.txt", "r");
    FILE *invertido = fopen("invertido.txt", "w");
    if (file == NULL || invertido == NULL) {
        printf("Erro ao abrir o arquivo.\n");
        return 1;
    }

    fseek(file, 0, SEEK_END);
    long tamanho = ftell(file);

    for (long i = tamanho - 1; i ⩾ 0; i--) {
        fseek(file, i, SEEK_SET);
        int c = fgetc(file);
        fputc(c, invertido);
    }

    fclose(file);
    fclose(invertido);
    return 0;
}
```

28.c

```c
#include <stdio.h>
#include <string.h>

typedef struct contato {
    char nome[50];
    char telefone[12];
    char email[50];
} Contato;

int main() {
    int opcao = 0;
    Contato contato;

    while (opcao != 4) {
        printf("\nDigite a opção desejada:\n");
        puts("1 - Cadastrar contato");
        puts("2 - Excluir contato");
        puts("3 - Listar contatos");
        puts("4 - Sair");
        scanf("%d", &opcao);
        getchar();

        if (opcao == 1) {
            FILE *agenda = fopen("agenda.txt", "a");
            if (agenda == NULL) {
                puts("Erro ao abrir agenda!");
                return 1;
            }

            puts("Digite o nome do contato");
            scanf("%s", contato.nome);
            puts("Digite o telefone do contato");
            scanf("%s", contato.telefone);
            puts("Digite o email do contato");
            scanf("%s", contato.email);

            fprintf(agenda, "%s;%s;%s\n", contato.nome,
contato.telefone, contato.email);
            fclose(agenda);

            puts("Cadastro realizado com sucesso!");
        }

        else if (opcao == 2) {
            FILE *agenda = fopen("agenda.txt", "r");
            FILE *temp = fopen("temp.txt", "w");
            if (agenda == NULL || temp == NULL) {
                puts("Erro ao abrir arquivos!");
                return 1;
```

```
            }

            char apagar[50];
            char linha[200];

            puts("Digite o nome do contato a ser apagado:");
            fgets(apagar, sizeof(apagar), stdin);
            apagar[strcspn(apagar, "\n")] = 0;

            while (fgets(linha, sizeof(linha), agenda)) {
                char copia[200];
                strcpy(copia, linha);

                for (int i = 0; copia[i]; i++) {
                    if (copia[i] >= 'A' && copia[i] <= 'Z')
                        copia[i] += ('a' - 'A');
                }

                if (strstr(copia, apagar) == NULL) {
                    fputs(linha, temp);
                }
            }

            fclose(agenda);
            fclose(temp);
            remove("agenda.txt");
            rename("temp.txt", "agenda.txt");
            puts("Contato apagado (se existia).");
        }

        else if (opcao == 3) {
            FILE *agenda = fopen("agenda.txt", "r");
            char linha[200];
            while (fgets(linha, sizeof(linha), agenda)) {
                printf("%s", linha);
            }
        } else {
            if (opcao != 4) {
                puts("Opcao invalida!");
            }
        }


    }

    return 0;
}
```

29.c

```c
#include <stdio.h>
#include <string.h>

typedef struct Produto {
 char id[20];
 char nome[50];
 float preco;
} Produto;

int main() {

    int escolha = 0;
    int descarte;

    while (escolha != 4) {
        printf("\nDigite a opção desejada:\n");
        puts("1 - Inserir produto");
        puts("2 - Listar produtos");
        puts("3 - Buscar por ID");
        puts("4 - Sair");
        scanf("%d", &escolha);
        getchar();

        if (escolha == 1) {
            FILE *banco = fopen("banco.bin", "ab");
            if (banco == NULL) banco = fopen("banco.bin", "w+b");

            Produto inserir, p;

            puts("Digite o nome do produto: ");
            if (fgets(inserir.nome, sizeof(inserir.nome), stdin) !=
NULL) {
                inserir.nome[strcspn(inserir.nome, "\n")] = 0;
            }

            puts("Digite o ID do produto: ");
            if (fgets(inserir.id, sizeof(inserir.id), stdin) != NULL)
{
                inserir.id[strcspn(inserir.id, "\n")] = 0;
            }

            puts("Digite o preço do produto: ");
            scanf("%f", &inserir.preco);
            getchar();

            int encontrado = 0;

            while (fread(&p, sizeof(Produto), 1, banco)) {
                if (strcmp(p.id, inserir.id) == 0) {
```

```
                    fseek(banco, -sizeof(Produto), SEEK_CUR);
                    fwrite(&inserir, sizeof(Produto), 1, banco);
                    encontrado = 1;
                    break;
                }
            }

            if (!encontrado) {
                fwrite(&inserir, sizeof(Produto), 1, banco);
            }

            fclose(banco);
            puts("Produto foi inserido com sucesso");
        }

        if (escolha == 2) {
            FILE *banco = fopen("banco.bin", "rb");
            Produto busca;
            if (banco == NULL) {
                puts("Erro ao abrir arquivo");
                return 1;
            }

            while (fread(&busca, sizeof(Produto), 1, banco)) {
                printf("ID: %s, Nome: %s, Preco: %f\n", busca.id,
busca.nome, busca.preco);
            }
        }

        if (escolha == 3) {
            FILE *banco = fopen("banco.bin", "rb");
            if (banco == NULL) {
                puts("Erro ao abrir arquivo");
                return 1;
            }

            char id[20];
            Produto p;
            puts("Digite o id a ser buscado:");
            if (fgets(id, sizeof(id), stdin) != NULL) {
                id[strcspn(id, "\n")] = 0;
            }

            int encontrado = 0;

            while (fread(&p, sizeof(Produto), 1, banco)) {
                if (strcmp(p.id, id) == 0) {
                    fseek(banco, -sizeof(Produto), SEEK_CUR);
                    printf("Produto encontrado!\n");
                    printf("ID: %s, Nome: %s, Preco: %f\n", p.id,
```

```
p.nome, p.preco);
                    encontrado = 1;
                    break;
                }
            }
        } else {
            if (escolha != 4) {
                puts("Opcao invalida!");
            }
        }


    }




    return 0;
}
```

30.c

```
#include <stdio.h>
#include <string.h>

typedef struct notas {
    char nome[50];
    char conteudo[500];
} Notas;

int main() {
    int opcao = 0;
    Notas nota;

    while (opcao != 4) {
        printf("\nDigite a opção desejada:\n");
        puts("1 - Criar nota");
        puts("2 - Excluir nota");
        puts("3 - Listar notas");
        puts("4 - Sair");
        scanf("%d", &opcao);
        getchar();

        if (opcao == 1) {
            FILE *notas = fopen("nota.txt", "a");
            if (notas == NULL) {
                puts("Erro ao abrir nota!");
                return 1;
            }

            puts("Digite o nome da nota: ");
```

```c
            if (fgets(nota.nome, sizeof(nota.nome), stdin) != NULL) {
                nota.nome[strcspn(nota.nome, "\n")] = 0;
            }
            puts("Digite o conteudo: ");
            if (fgets(nota.conteudo, sizeof(nota.conteudo), stdin) !=
NULL) {
                nota.conteudo[strcspn(nota.conteudo, "\n")] = 0;
            }


            fprintf(notas, "%s;%s\n", nota.nome, nota.conteudo);
            fclose(notas);

            puts("Nota salva com sucesso!");
        }

        else if (opcao == 2) {
            FILE *notas = fopen("nota.txt", "r");
            FILE *temp = fopen("temp.txt", "w");
            if (notas == NULL || temp == NULL) {
                puts("Erro ao abrir arquivos!");
                return 1;
            }

            char apagar[50];
            char linha[200];

            puts("Digite o nome do nota a ser apagado:");
            fgets(apagar, sizeof(apagar), stdin);
            apagar[strcspn(apagar, "\n")] = 0;

            while (fgets(linha, sizeof(linha), notas)) {
                char copia[200];
                strcpy(copia, linha);

                for (int i = 0; copia[i]; i++) {
                    if (copia[i] >= 'A' && copia[i] <= 'Z')
                        copia[i] += ('a' - 'A');
                }

                if (strstr(copia, apagar) == NULL) {
                    fputs(linha, temp);
                }
            }

            fclose(notas);
            fclose(temp);
            remove("nota.txt");
            rename("temp.txt", "nota.txt");
            puts("nota apagado (se existia).");
```

```c
        }

        else if (opcao == 3) {
            FILE *notas = fopen("nota.txt", "r");
            if (notas == NULL) {
                puts("Nenhuma nota encontrada.");
                continue;
            }

            char linha[600];
            int contador = 1;

            puts("\n==== LISTA DE NOTAS ====\n");

            while (fgets(linha, sizeof(linha), notas)) {
                linha[strcspn(linha, "\n")] = 0;

                char *nome = strtok(linha, ";");
                char *conteudo = strtok(NULL, ";");

                if (nome && conteudo) {
                    printf("Nota %d\n", contador++);
                    printf("Título: %s\n", nome);
                    printf("Conteúdo: %s\n", conteudo);
                    printf("------------------------------\n");
                }
            }

            if (contador == 1)
                puts("Nenhuma nota encontrada.");

            fclose(notas);
        }


    }

    return 0;
}
```