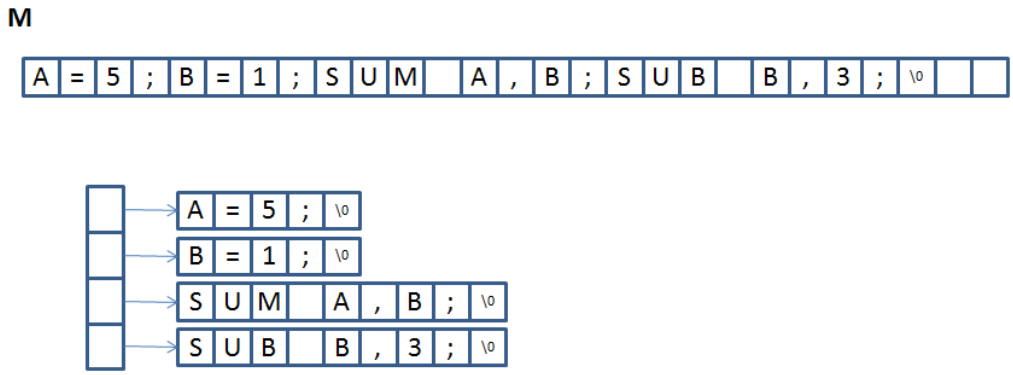
Universidade Federal de Ouro Preto - UFOP
Disciplina: BCC 201 - Introdução à Programação
Professor: Guillermo Cámara-Chávez

Aluno: _____ **No.** _____

A cola não será tolerada. Se alguém for pego colando, será reprovado com Zero. É considerado cola: olhar/copiar da prova de outro ou deixar outro aluno olhar sua prova.

3ra. Avaliação - Grupo B

1. (2 pts) Seja M um vetor unidimensional de caracteres que representa a memória de um computador. Nela encontram-se as diferentes instruções de um algoritmos, casa instrução finaliza com ';'. Crie uma função que permita retornar um vetor de *strings*, onde casa posição do vetor contem um única instrução.



```
char** Intrucoes(char* M, int *n)
{
    // ENTRADA
    // M: memoria
    // n: representa o numero total de instrucoes, sera
    //    contado dentro desta funcao
    // SAIDA
    // V: vetor com as intrucoes
    int i, j = 0, tam = strlen(M), k;
    char **V, tmp[50];
    *n = 0;
    for (i = 0; i < tam; i++)
        if (M[i] == ';') (*n)++;
    V = (char**) calloc(*n, sizeof(char*));
    for (i = 0; i < *n; i++)
    {
        k = 0;
        while(M[j] != ';')
            tmp[k++] = M[j++];
        j++;
        tmp[k++] = ';';
        tmp[k] = '\0';
        V[i] = (char*) calloc(strlen(tmp)+1, sizeof(char));
        strcpy(V[i], tmp);
    }
    return V;
}
```

2. (2 pts) Foi escrito um arquivo binário com os dados de n pessoas. O arquivo foi gerado a partir dos dados definidos na estrutura *Pessoa* com os campos: *nome* e *idade*. Ordene alfabeticamente pelo campo nome os dados escritos no arquivo. Faça a ordenação diretamente no arquivo. DICA: use a função `fseek()` junto com parâmetro `SEEK_CUR` para se movimentar dentro do arquivo.

```
typedef struct Aluno{
    char nome[50];
    int idade;
}TAluno;

void Ordena(char* nomeArquivo)
{
    FILE* pf = fopen(nomeArquivo, "r+b");
    int i, j, n = 0;
    TAluno tmpA, tmpB;
    while(fread(&tmpA, sizeof(TAluno), 1, pf) != 0)
        n++;
    rewind(pf);
    for (i = 0; i < n; i++)
    {
        for (j = 1; j < n; j++)
        {
            fread(&tmpA, sizeof(TAluno), 1, pf);
            fread(&tmpB, sizeof(TAluno), 1, pf);
            if (strcmp(tmpA.nome, tmpB.nome) > 0)
            {
                fseek(pf, -2*sizeof(TAluno), SEEK_CUR);
                fwrite(&tmpB, sizeof(TAluno), 1, pf);
                fwrite(&tmpA, sizeof(TAluno), 1, pf);
            }
            fseek(pf, -sizeof(TAluno), SEEK_CUR);
        }
        rewind(pf);
    }
    fclose(pf);
}
```

3. (2 pts) Faça um programa que solicite o nome de um arquivo e conte quantas palavras com 2, 3, 4, 5 e mais de 5 letras que existem no arquivo.

```
void ContaPalavras(char* ArquivoNome)
{
    FILE* pf = fopen(ArquivoNome, "r");
    if (pf == NULL) return;
    char palavra[50];
    int i, conta[6] = {0};
    while(fscanf(pf, "%s", palavra) != EOF)
    {
        switch(strlen(palavra))
        {
        case 1: conta[0]++; break;
        case 2: conta[1]++; break;
        case 3: conta[2]++; break;
        case 4: conta[3]++; break;
        case 5: conta[4]++; break;
        default: conta[5]++;
        }
    }
    for (i = 0; i < 6; i++)
        printf("Palavras com %d letras: %d \n", i+1, conta[i]);
    fclose(pf);
}
```

4. (1 pt) Seja o seguinte trecho de programa:

```
int i = 3, j = 5;
int *p, *q;
p = &i;
q = &j;
```

Qual é o valor das seguinte expressões?

(a) $*p - *q = 3 - 5$

(b) $**\&p = 3$

5. (1 pt) Qual será a saída deste programa?

```
int main(){
    int i = 5, *p = &i;
}
```

Qual é o valor das seguintes expressões: (indicar se alguma expressão é invalida)

(a) $p;$ = endereço de i

(b) $*p + 2; = 5 + 2$

(c) $**\&p; = 5$

(d) $3 * *p; = 3 * 5$

(e) $**\&p + 4; = 5 + 4$

6. (2 pts) Elaborar um algoritmo que auxilie no controle de uma fazenda de gado que possuí um total de 2000 cabeças de gado. A base de dados é formada por um conjunto de estruturas (registros) contendo os seguintes campos referente a cada cabeça de gado:

   (a) código: código da cabeça de gado,

   (b) leite: número de litros de leite produzido por semana,

   (c) alim: quantidade de alimento ingerida por semana - em quilos,

   (d) nasc: data de nascimento - mês e ano,

   (e) abate: 'N' (não) ou 'S' (sim).

   O campo nasc. é do tipo struct data que por sua vez, possui dois campos: mês e ano Elaborar funções para:

   (a) Ler a base de dados (código, leite, alim, nasc.mês e nasc.ano), armazenado em um vetor de estruturas.

   (b) Preencher o campo abate, considerando que a cabeça de gado irá para o abate caso:

      - tenha mais de 5 anos, ou;
      - produza menos de 40 litros de leite por semana, ou;
      - produza entre 50 e 70 litros de leite por semana e ingira mais de 50 quilos de alimento por dia

   Criar a função que devolva a quantidade total de leite que vai ser produzido por semana na fazenda, após o abate

```c
typedef struct Data{
    int mes, ano;
}TData;
typedef struct Gado
{
    int codigo, leite, alim;
    TData nasc;
    char abate;
}TGado;

void Inserir(TGado* V, int n)
{
    int i;
    for (i = 0; i < n; i++)
    {
        printf("Digite codigo: ");
        scanf("%d", &V[i].codigo);
        printf("Digite numero de litros: ");
        scanf("%d", &V[i].leite);
        printf("Digite quantidade de alimento: ");
        scanf("%d", &V[i].alim);
        printf("Digite data (mes e ano): ");
        scanf("%d %d", &V[i].nasc.mes, &V[i].nasc.ano);

        if ( (2014 - V[i].nasc.ano > 5) || (V[i].leite < 40) ||
             (V[i].leite >= 50 && V[i].leite <= 70 && V[i].alim > 50) )
            V[i].abate = 's';
        else
            V[i].abate = 'n';
    }
}

int ProducaoLeite(TGado* V, int n)
{
    int i, litros = 0;
    for (i = 0; i < n; i++)
        if (V[i].abate == 'n')
            litros += V[i].leite;
    return litros;
}
```