ALUNO: VITOR BRUNO DE OLIVEIRA BARTH

1 - Faça um programa que leia 10 conjuntos de dois valores, o primeiro representando o número do aluno e o segundo representando a sua altura em centímetros. Encontre o aluno mais alto e o mais baixo. Mostre o número do aluno mais alto e o número do mais baixo junto com suas alturas.

```
{       10 conjuntos de 2 valores
        Um representa o numero do aluno e outro representa a altura do aluno
        Mostre o número do aluno mais alto e do mais baixo junto com a altura  }

algoritmo
        { declare variaveis }
        defina  ALUNOS 10;
        declare numero, altura, maiorNumero, maiorAltura, menorNumero, menorAltura, i (numérico)

        { leia primeiro aluno }
        leia  maiorNumero
        menorNumero <- maiorNumero
        leia  maiorAltura
        menorAltura <- maiorAltura
        i <- 1;

        { loop leia alunos e verificar condições }
        faça
                leia numero
                leia altura
                se (altura > maiorAltura) então
                        maiorNumero <- numero
                        maiorAltura <- altura
                fim-se
                se (altura < menorAltura) então
                        menorNumero <- numero
                        menorAltura <- altura
                fim-se
                i++
        enquanto (i < ALUNOS)

        { imprima resultado }
        imprima maiorNumero e maiorAltura
        imprima menorNumero e menorAltura
fim-algoritmo
```

```c
#include <stdio.h>
#define ALUNOS 10

int main() {
	// Declarar variaveis
	int num, altura, maiorNum, maiorAltura, menorNum, menorAltura, i;

	// Ler o primeiro aluno
	printf("Numero do aluno 1: ");
	scanf("%i", &maiorNum);
	printf("Altura do aluno 1 (em cm): ");
	scanf("%i", &maiorAltura);
	menorNum = maiorNum;
	menorAltura = maiorAltura;
	i = 2;

	// Loop e verificar condicoes
	do {
		printf("Numero do aluno %i: ", i);
		scanf("%i", &num);
		printf("Altura do aluno %i (em cm): ", i);
		scanf("%i", &altura);

		if (altura > maiorAltura) {
			maiorNum = num;
			maiorAltura = altura;
		}

		if (altura < menorAltura) {
			menorNum = num;
			menorAltura = altura;
		}

		i++;
	} while (i <= ALUNOS);

	// Imprimir os resultados
	printf("Maior aluno: Numero %i, Altura %i", maiorNum, maiorAltura);
	printf("\nMenor aluno: Numero %i, Altura %i", menorNum, menorAltura);

	return 0;
}
```

2 - Foi feita a estatística em 5 cidades brasileiras para coletar dados sobre acidentes de trânsito. Foram obtidos os seguintes dados: a) código da cidade; b) número de veículos de passeio; c) número de acidentes de trânsito com vítimas; Deseja-se saber: a) qual o maior e menor índice de acidentes de trânsito e a que cidades pertencem; b) qual a média de veículos nas cinco cidades juntas; c) qual a média de acidentes de trânsito nas cidades com menos de 2000 veículos de passeio.

```
{       5 conjuntos de 3 valores
        Codigo da cidade, numero de veiculos de passeio, numero de acidentes com vitimas
        Resulta o maior e menor indice de acidentes e as cidades
        A media de veiculos nas cidades juntas
        A media de acidentes nas cidades com menos de 2000 veiculos }

algoritmo
        { declare variaveis }
        defina  COMPARADOR 2000;
        defina  NUMCIDADES 5;
        declare codigo, numVeiculos, numAcidentes, maiorAcidentes, maiorCodigo, menorAcidentes, menorCodigo, medVeiculos,
medAcidentes, i, j (numérico)

        { leia primeira cidade }
        leia maiorCodigo
        menorCodigo <- maiorCodigo
        leia numVeiculos
        medVeiculos <- numVeiculos
        leia maiorAcidentes
        menorAcidentes <- maiorAcidentes
        se (numVeiculos < COMPARADOR)
                medAcidentes <- numVeiculos
                i <- 1
        fim-se
        j <- 1

        { loop leia codigo da cidade, num de veiculos e num de acidentes }
        faça
                leia codigo
                leia numVeiculos
                medVeiculos <- medVeiculos + numVeiculos
                leia numAcidentes
                se (numVeiculos < 2000)
                        medAcidentes <- medAcidentes + numVeiculos
                        i++
                fim-se
                se (numAcidentes > maiorAcidentes) então
                        maiorAcidentes <- numAcidentes
                fim-se
                se (numAcidentes < menorAcidentes) então
                        menorAcidentes <- numAcidentes
                fim-se
                j++
        enquanto (j < NUMCIDADES)

        { imprima os resultados }
        imprima maiorCodigo e maiorAcidentes
        imprima menorCodigo e menorAcidentes
        imprima medVeiculos/NUMCIDADES
        imprima medAcidentes/i
fim-algoritmo
```

```c
#include <stdio.h>
#define NUMCIDADES 5
#define COMPARADOR 2000

int main() {
	// Declarar variaveis
	int cod, numVei, numAci, maiorCod, maiorAci, menorCod, menorAci, i, j;
	float medAci, medVei;

	// Ler primeira cidade
	printf("Insira o código da 1a cidade: ");
	scanf("%i", &maiorCod);
	printf("Insira o numero de veiculos da 1a cidade: ");
	scanf("%i", &medVei);
	printf("Insira o numero de acidentes 1a cidade: ");
	scanf("%i", &maiorAci);
	menorCod = maiorCod;
	menorAci = maiorAci;
	i = 2;

	// Verificar se cidade 1 possui mais de COMPARADOR veiculos
	if (medVei > COMPARADOR) {
		medAci = medVei;
		j = 1;
	}

	// Loop para ler as outras cidades e verificar COMPARADOR veiculos
	do {
		// Ler i cidade
		printf("Insira o código da %ia cidade: ", i);
		scanf("%i", &cod);
		printf("Insira o numero de veiculos da %ia cidade: ", i);
		scanf("%i", &numVei);
		printf("Insira o numero de acidentes %ia cidade: ", i);
		scanf("%i", &numAci);
		i++;
		medVei += numVei;

		// Verificar num acidentes
		if (numAci > maiorAci) {
			maiorAci = numAci;
			maiorCod = cod;
		}
		if (numAci < menorAci) {
			menorAci = numAci;
			menorCod = cod;
		}

		// Verificar se cidade 1 possui mais de COMPARADOR veiculos
		if (medVei > COMPARADOR) {
			medAci += numAci;
			j++;
		}
	} while (i <= NUMCIDADES);

	// Imprimir os resultados
	printf("\nMaior numero de acidentes: Código: %i, Num de Acidentes:%i", maiorCod, maiorAci);
	printf("\nMenor numero de acidentes: Código: %i, Num de Acidentes:%i", menorCod, menorAci);
	printf("\nMedia de veiculos: %f", medVei/NUMCIDADES);
	printf("\nMedia de acidentes em cidades com menos de %i veiculos: %f", COMPARADOR, medAci/j);
}
```

3 - Uma empresa possui 10 funcionários com as seguintes características: código, número de horas trabalhadas no mês, turno de trabalho (M-Matutino, V-Vespertino ou N-Noturno), categoria (O-Operário ou G-Gerente), valor da hora trabalhada. Sabendo-se que essa empresa deseja informatizar sua folha de pagamento, faça um programa que: a) Leia as informações dos funcionários, exceto o valor da hora trabalhada, não permitindo que sejam informados turnos nem categorias inexistentes. Trabalhar sempre com a digitação de letras maiúsculas. b) Calcule o valor da hora trabalhada, conforme tabela. Adote o valor de R$ 150,00 para o salário mínimo. c) Calcule o salário inicial dos funcionários com base no valor da hora trabalhada e no número de horas trabalhadas. d) calcule o valor do auxílio-alimentação recebido por funcionário de acordo com o seu salário inicial, conforme tabela. e) mostre o código, número de horas trabalhadas, valor da hora trabalhada, salário inicial, auxílio alimentação e o salário final (salário inicial e auxílio alimentação).

```
{       10 funcionarios com codigo, numero de horas trabalhadas, turno de trabalho, categoria, valor da hora trabalhada
        leia as informacoes (exceto valor da hora trabalhada), não permitindo categorias inexistentes.
        TUDO EM LETRA MAIUSCULA
        Calcule o salario inicial com base no valor da hora trabalhada
        Calcule o auxilio alimentacao
        Mostre todas as informações         }

algoritmo
        { declara as variaveis }
        defina  NFUNCIONARIOS 10
        defina  SALARIOMINIMO 150
        defina tMatriz <- NFUNCIONARIOS-1
        declare codigo[tMatriz], nHoras[tMatriz], turno[tMatriz], vHora[tMatriz], salario[tMatriz], auxilio[tMatriz] (numérico)
        declare categoria[tMatriz],  turno[tMatriz] (literal)

        { loop leia dados funcionarios }
        para i de 0 a tMatriz faça
                leia codigo[i]
                leia nHoras[i]
                leia turno[i]
                { impedir valores invalidos }
                enquanto (turno[i] != 'M' || turno[i] != 'V' || turno[i] != 'N') faça
                        imprima "Digite um turno válido"
                        leia  turno[i]
                fim-enquanto
                leia categoria[i]
                { impedir valores invalidos }
                enquanto (categoria[i] != 'O' || categoria[i] != 'G') faça
                        imprima "Digite um cargo válido"
                        leia turno[i]
                fim-enquanto

                { calcular os valores da hora trabalhada }
                se (categoria[i] == 'G' && turno == 'N') então
                        vHora <- ((SALARIOMINIMO/100)*18)
                se (categoria[i] == 'G' && (turno == 'M' || turno[i] == 'V')) então
                        vHora <- ((SALARIOMINIMO/100)*15)
                se (categoria[i] == 'O' && turno == 'N') então
                        vHora <- ((SALARIOMINIMO/100)*13)
                se (categoria[i] == 'O' && (turno == 'M' || turno[i] == 'V')) então
                        vHora <- ((SALARIOMINIMO/100)*10)

                { calcular os valor do salario }
                salario[i] <- nHoras*vHora;

                { calcular os valores da alimentacao }
                se (salario < 300) então
                        alimentacao <- ((salario/100)*20)
                se (salario >= 300 && salario <= 600) então
                        alimentacao <- ((salario/100)*15)
                se (salario > 600) então
                        alimentacao <- ((salario/100)*5)
        fim-para

        { imprima os dados }
        para i de 0 a tMatriz faça
                imprima codigo[i]
                imprima nHoras[i]
                imprima vHora[i]
                imprima salario[i]
                imprima alimentacao[i]
                imprima salario[i]+alimentacao[i]
        fim-para
fim-algoritmo
```

```c
#include <stdio.h>
#define NUMFUNCIONARIOS 10
#define SALARIOMIN 150

int main() {
	int M = NUMFUNCIONARIOS-1;
	int codigo[M], nHoras[M], turno[M], vHora[M], salario[M], auxilio[M]
	char categoria[M], turno[M];

	// Loop ler dados dos funcionarios
	for (int i = 0; i < NUMFUNCIONARIOS; i++) {
		printf("Insira o código do funcionário: ");
		scanf("%i", &codigo[i]);
		printf("Insira o número de horas trabalhadas: ");
		scanf("%i", &nHoras[i]);
		printf("Insira o turno (M, V, N): ");
		scanf("%s", &turno[i]);

		// Impedir valores inválidos para turno
		while (turno[i] != 'M' || turno[i] != 'V' || turno[i] != 'N') {
			printf("Insira um turno válido!");
			printf("Insira o turno (M, V, N): ");
			scanf("%s", &turno[i]);
		}

		printf("Insira a categoria (G ou O): ");
		scanf("%s", &categoria[i]);

		// Impedir valores invalidos para categoria
		while (categoria[i] != 'O' || categoria[i] != 'G') {
			printf("Insira uma categoria válida!");
			printf("Insira a categoria (G ou O): ");
			scanf("%s", &categoria[i]);
		}

		// Calcular o valor da hora trabalhada
		if (categoria[i] == 'G' && turno == 'N')
			vHora = ((SALARIOMIN/100)*18)
		if (categoria[i] == 'G' && (turno == 'M' || turno[i] == 'V'))
			vHora = ((SALARIOMIN/100)*15)
		if (categoria[i] == 'O' && turno == 'N')
			vHora = ((SALARIOMIN/100)*13)
		if (categoria[i] == 'O' && (turno == 'M' || turno[i] == 'V'))
			vHora = ((SALARIOMIN/100)*10)

		// Calcular os valores do salario
		salario[i] = nHoras*vHora;

		// Calcular os valores da alimentacao
		if (salario[i] < 300)
			alimentacao[i] = ((salario[i]/100)*20);
		if (salario[i] >= 300 && salario[i] <= 600)
			alimentacao[i] = ((salario[i]/100)*15);
		if (salario[i] > 600)
			alimentacao[i] = ((salario[i]/100)*5);
	}

	// Imprimir os dados
	for (int i =0; i < NUMFUNCIONARIOS, i++) {
		printf("\nCodigo: %i", codigo[i]);
		printf("\nNúmero de horas trabalhadas: %i", nHoras[i]);
		printf("\nValor da hora trabalhador: %i", nHoras[i]);
		printf("\Salario: %i", salario[i]);
		printf("\nAlimentação: %i", alimentacao[i]);
		printf("\nTotal a Pagar: %i", salario[i]+alimentacao[i]);
	}
}
```

4 - Faça um programa que monte os 20 primeiros termos da sério de Fibonacci.

```
{ os 20 primeiros termos de fibonacci }

algoritmo
        defina  NTERMOS 20;
        declare termoAnt, termoM, termoProx (numérico)
        termoAnt <- 0
        termoProx <- 1

        para i de 0 a NTERMOS-1
                imprima termoAnt
                termoM <- termoAnt
                termoAnt <- termoProx
                termoProx <- termoM + termoProx
fim-algoritmo
```

```c
// Os 20 primeiros termos de fibonacci

#include <stdio.h>
#define NTERMOS 20

int main() {
	int termoAnt, termoM, termoProx;
	termoAnt = 0;
	termoProx = 1;

	for (int i = 0; i < NTERMOS; i++) {
      printf("%i, ", i, termoAnt);
			termoM = termoAnt;
			termoAnt = termoProx;
			termoProx = termoM + termoProx;
   }

   return 0;
}
```

5 - Uma pesquisa sobre algumas características físicas da população de uma determinada região coletou os seguintes dados, referentes a cada habitante, para serem analisados: ● sexo (masculino, feminino); ● cor dos olhos (Azuis, Verdes, Castanhos); ● cor dos cabelos (louros, castanhos, pretos); ● idade em anos. Para cada habitante, foi digitada uma linha com esses dados e a última linha, que não corresponde a ninguém, conterá o valor de idade igual a -1. Fazer um algoritmo que determine e escreva: a) a maior idade dos habitantes; b) a porcentagem de indivíduos do sexo feminino cuja idade está entre 18 e 35 anos inclusive e que tenham olhos verdes e cabelos castanhos e cabelos louros.

```
{       leia  sexo, cor dos olhos, cabelos, idade
        ultima linha = -1
        escreva a maior idade
        escreva porcentagem entre 18 e 35 anos inclusive, que tenham olhos verdes, cabelos castanhos OU cabelos louros
        PS: Professor, eu não entendi o enunciado do exercício, portanto fiz de acordo com o comentário acima          }

lgoritmo
        defina VULTIMO -1;
        declare idade, mIdade (numérico)
        declare cOlhos, cCabelos, sexo, pVerdes, pCastanhos, pLouros (literal)

        { leia o primeiro }
        leia  idade
        mIdade <- idade
        leia  cCabelos
        leia  cOlhos
        leia  sexo
        se (idade >= 18 && idade <= 35 && (sexo == 'F' || sexo == 'f')) então
                se (cOlhos == 'V' || cOlhos == 'v') então
                        pVerdes <- 1
                fim-se
                se (cCabelos == 'C' || cCabelos == 'c') então
                        pCastanhos <- 1
                fim-se
                se (cCabelos == 'L' || cCabelos == 'l') então
                        pLouros <- 1
                fim-se
        fim-se
        i <- 1

        { loop leitura }
        enquanto (idade != VULTIMO) faça
                leia  idade
                se (idade != VULTIMO) então
                        se (idade > mIdade) então
                                mIdade <- idade
                        fim-se
                        leia  cCabelos
                        leia  cOlhos
                        leia  sexo
                        se (idade >= 18 && idade <= 35 && (sexo == 'F' || sexo == 'f')) então
                                se (cOlhos == 'V' || cOlhos == 'v') então
                                        pVerdes <- 1
                                fim-se
                                se (cCabelos == 'C' || cCabelos == 'c') então
                                        pCastanhos <- 1
                                fim-se
                                se (cCabelos == 'L' || cCabelos == 'l') então
                                        pLouros <- 1
                                fim-se
                        fim-se
                        i <- i+1
                fim-se
        fim-enquanto

        imprima mIdade
        imprima (pVerdes/i)*100
        imprima (pCastanhos/i)*100
        imprima (pLouros/i)*100
fim-algoritmo
```

```c
#include <stdio.h>
#define VULTIMO -1

int main() {
        int idade, mIdade, i;
        char cOlhos, cCabelos, sexo;
   float  pVerdes, pCastanhos, pLouros;

        // leia o primeiro
   printf("Digite idade = -1 para sair \n");
   printf("Insira a idade de 1:" );
        scanf(" %i", &idade);
   mIdade = idade;
   pVerdes = 0;
   pCastanhos = 0;
   pLouros = 0;
   if (idade != VULTIMO) {
        printf("Insira a cor dos cabelos de 1 (C, L ou P):" );
        scanf(" %c", &cCabelos);
        printf("Insira a cor dos olhos de 1 (A, V ou C):" );
        scanf(" %c", &cOlhos);
        printf("Insira o sexo de 1 (M ou F):" );
        scanf(" %c", &sexo);
        i = 1;
        if ((idade >= 18 && idade <= 35) && (sexo == 'F' || sexo == 'f')) {
           if (cOlhos == 'V' || cOlhos == 'v')
              pVerdes = 1;

           if (cCabelos == 'C' || cCabelos == 'c')
              pCastanhos = 1;

           if (cCabelos == 'L' || cCabelos == 'l')
              pLouros = 1;
        }
    }

        // loop leitura
        while (idade != VULTIMO) {
                i++;
                printf("Insira a idade de %i:", i );
                scanf("%i", &idade);
                if (idade != VULTIMO) {
                        if (idade > mIdade)
                                mIdade = idade;
                        printf("Insira a cor dos cabelos de %i: (C, L ou P):", i );
        scanf(" %c", &cCabelos);
        printf("Insira a cor dos olhos de %i (A, V ou C):", i );
        scanf(" %c", &cOlhos);
        printf("Insira o sexo de %i (M ou F):", i);
        scanf(" %c", &sexo);
                        if ((idade >= 18 && idade <= 35) && (sexo == 'F' || sexo == 'f')) {
           if (cOlhos == 'V' || cOlhos == 'v')
              pVerdes++;

           if (cCabelos == 'C' || cCabelos == 'c')
              pCastanhos++;

           if (cCabelos == 'L' || cCabelos == 'l')
              pLouros++;
        }
      }
   }

    pVerdes = (pVerdes/i)*100);
    pCastanhos = (pCastanhos/i)*100);
    pLouros = (pLouros/i)*100;

        printf("\nA maior idade: %i", mIdade);
        printf("\nA porcentagem de mocas de 18 a 35 anos com olhos verdes e: %f", pVerdes);
        printf("\nA porcentagem de mocas de 18 a 35 anos com cabelos castanhos e: %f:", pCastanhos);
        printf("\nA porcentagem de mocas de 18 a 35 anos com cabelos louros e: %f", pLouros);

   return 0;
}
```

6 - Num frigorífico existem 90 bois. Cada boi traz preso em seu pescoço um cartão contendo seu número de identificação e seu peso. Fazer um algoritmo que escreva o número e o peso do boi mais gordo e do boi mais magro.

```
{       Ler 90 valores
        Numero de Indet if icacao e Peso do Boi
        Escrever o numero e o peso do mais gordo e do mais magro }

algoritmo
        { declare variaveis }
        defina NUMBOIS 90
        declare i, peso, ident if icacao, maiorId, maiorPeso, menorId, menorPeso (numérico)

        { leia o primeiro }
        leia  maiorId
        menorId <- maiorId
        leia  maiorPeso
        menorPeso <- menorId

        { fazer o loop }
        faça
                leia ident if icacao
                leia peso
                se (peso > maiorPeso) então
                        maiorPeso <- peso
                        maiorId <- ident if icacao
                fim-se
                se (peso < menorPeso) então
                        menorPeso <- peso
                        menorId <- ident if icacao
                fim-se
                i++
        enquanto (i < NUMBOIS)

        imprima maiorId, maiorPeso
        imprima menorId, menorPeso
fim-algoritmo
```

```c
#include <stdio.h>
#define NUMBOIS 90

int main() {
// declare variaveis
   int i, identificacao, maiorId, menorId;
   float peso, maiorPeso, menorPeso;

// leia o primeiro
   printf("Insira a identificacao de 1: ");
            scanf("%i", &maiorId);
            menorId = maiorId;
            printf("Insira o peso de 1: ");
   scanf("%f", &maiorPeso);
            menorPeso = maiorPeso;
            i = 1;
            // Loop
            do {
      i++;
      printf("Insira a identificacao de %i: ", i);
      scanf("%i", &identificacao);

      printf("Insira o peso de %i: ", i);
      scanf("%f", &peso);

      if (peso > maiorPeso) {
         maiorId = identificacao;
         maiorPeso =  peso;
      }

      if (peso < menorPeso) {
         menorId = identificacao;
         menorPeso =  peso;
      }
   } while (i < NUMBOIS);

printf("\nO maior peso é do boi %i: %f Kg", maiorId, maiorPeso);
   printf("\nO menor peso é do boi %i: %f Kg", menorId, menorPeso);
}
```

7 - Fazer um algoritmo que: ● leia um número indeterminado de linhas contendo cada uma a idade de um indivíduo. A última linha, que não entrará nos cálculos, contém o valor da idade igual a zero. ● Calcule e escreva a idade média deste grupo de indivíduos.

```
algoritmo
        {       Ler a idade de individuos até digitar 0;
                Escrever a idade media do grupo         }
        declare mediaidade, idade, i (numérico)

        { Ler o primeiro }
        leia idade;
        mediaidade <- 0
        i <- 0

        { Loop de leitura }
        enquanto (idade != 0) faça
                mediaidade <- mediaidade + idade
                i++
                leia idade
        fim-enquanto

        { Imprima o resultado }
        imprima mediaidade/i
fim-algoritmo
```

```
/*      Ler a idade de individuos até digitar 0
                Escrever a idade media do grupo                 */

#include <stdio.h>

int main() {
        int idade, i;
   float mediaidade;
        // Ler o primeiro
   printf("Insira a idade de 1: ");
   scanf("%i", &idade);
        mediaidade = 0;
        i = 0;

        // Loop de leitura
        while (idade != 0) {
      mediaidade = mediaidade + idade;
      i++;
      printf("Insira a idade de %i: ", i);
      scanf("%i", &idade);
   }

   mediaidade = mediaidade/i;
        printf("\nA média das idades é %f", mediaidade);

   return 0;
}
```

8 - Tem-se um conjunto de dados contendo a altura e o sexo (masculino, feminino) de 50 pessoas. Fazer um algoritmo que calcule e escreva: - a maior e a menor altura do grupo; - a média de altura das mulheres; - o número de homens.

```
{ Esse algoritmo lê Altura, Sexo de um grupo de 50 pessoas fornece como resultado:
                - Maior e menor altura
                - Menor altura de mulheres
                - O número de homens                }

algoritmo
        { Declaração de variáveis }
        defina  QUANTIDADE: 50
        declare menor, media, maior, mulheres, i,  altura (numérico)
        declare sexo (literal)

        { Leitura da primeira pessoa }
        leia  altura, sexo

        { Inicialização das variáveis }
        menor <- altura
        maior <- altura
        media <- 0
        homens <- 0

        se  (sexo = 'f' ou sexo = 'F') então
                mulheres <- 1
                media <- altura
        fim-se

        { Laço de repetição para acumular as alturas lidas, o número de homens e determinar a maior e menor alturas }
        para i de 1 até 49 faça
                leia  altura, sexo
                { determinar maior e menor }
                se (altura < menor) então
                        menor <- altura
                fim-se
                se (altura > maior) então
                        maior <- altura
                fim-se

                { Acumular quantidade de homens e alturas lidas }
                se sexo = 'f' ou sexo = 'F' então
                        mulheres <- mulheres + 1
                        media <- media + altura
                fim-se
        fim-para

        { imprima os resultados }
        imprima menor, maior, (media/mulheres), (QUANTIDADE-mulheres)
fim-algoritmo
```

```
/* Esse algoritmo lê Altura, Sexo de um grupo de 50 pessoas fornece como resultado:
                - Maior e menor altura
                - Menor altura de mulheres
                - O número de homens */

#include <stdio.h>
#define QUANTIDADE 50

int main() {
        // Declaração de variáveis

        float menor, media, maior, mulheres, i, altura;
        char sexo;

        // Leitura da primeira pessoa
        printf("Entre com o valor da altura em cm ");
        scanf("%d", &altura);
        printf("\nEntre com o valor do sexo (m e f) ");
        scanf("%s", &sexo);

        // Inicialização das variáveis
        menor = altura;
        maior = altura;
        media = 0;
        mulheres = 0;

        if (sexo == 'f' || sexo == 'F') {
                mulheres = 1;
                media = altura;
        }

        if  (altura < menor) {
                        menor = altura;
                }

                if (altura > maior) {
                        maior = altura;
        }

        // Laço de repetição para acumular as alturas lidas, o número de homens e determinar a maior e menor alturas
        for (i = 1; i < QUANTIDADE; i++) {
                printf("\nEntre com o valor da altura em cm ");
                scanf("%i", &altura);
                printf("\nEntre com o valor do sexo (m e f) ");
                scanf("%s", &sexo);

                // Determinar maior e menor
                if  (altura < menor) {
                        menor = altura;
                }

                if (altura > maior) {
                        maior = altura;
                }


                // Acumular quantidade de homens e alturas lidas
                if (sexo == 'f' || sexo == 'F') {
                        mulheres += 1;
                        media = altura;
                }
        }
        // Imprima o resultados


        printf("\n O menor valor é %f \n O maior valor é %f \n A media da altura das mulheres é %f \n A quantidade de homens é
%f", menor, maior, media/mulheres, mulheres);

        return 0;
}
```

9 - Dados dois números reais e um caracter (+,-,*,/) representando uma operação a ser efetuada com eles, calcule e informe o resultado da operação.

```
{ dado dois numeros e uma operação, calcule o resultado }

algoritmo
        { declare variaveis }
        declare n1, n2 (numero)
        declare op (literal)

        { leia os valores }
        leia  n1
        leia  op
        leia  n2

        { imprima }
        se (op == '+') então
                imprima n1+n2
        fim-se
        se (op == '-') então
                imprima n1-n2
        fim-se
        se (op == '*') então
                imprima n1*n2
        fim-se
        se (op == '/') então
                imprima n1/n2
        fim-se
fim-algoritmo
```

```c
#include <stdio.h>

int main() {
	float n1, n2;
	char op;

	printf ("\n insira n1:");
	scanf ("%f", &n1);
	printf ("\n insira n2:");
	scanf ("%f", &n2);
	printf ("n\ insira operação:");
	scanf ("%s", &op);

	if (op== "+")
		printf ("%f", n1+n2);
	if (op== "-")
		printf ("%f", n1-n2);
	if (op== "*")
		printf ("%f", n1*n2);
	if (op== "/")
		printf ("%f", n1/n2);

	return 0;
}
```

10 - O perfil de uma pessoa é dado pelo seu ano de nascimento. Por exemplo, se o ano é 1987, calculamos a soma 19 + 87, dividimos o seu resultado (106) por 5 e pegamos o resto (1). Este resto indica o perfil da pessoa: 0 - tímido; 1 – sonhador(a); 2 – paquerador(a); 3 – atraente; 4 – irresistível; Dado o ano de nascimento de uma pessoa, informe qual o seu perfil.

```
{       leia  o ano de nascimento da pessoa, divida em dois, some ambas as partes por 5 e pegue o resto
        dependendo do resto, mostre uma resposta }

algoritmo
        { declare as variaveis }
        declare ano, v1, v2, resultado (numérico)

        { leia o ano }
        leia ano
        v1 <- ano/100
        v2 <- ano - v1*100

        { formar o resultado e interpretar }
        resultado <- (v1+v2)%5
        se (resultado == 0) então
                imprima "tímido(a)"
        fim-se
        se (resultado == 1) então
                imprima "sonhador(a)"
        fim-se
        se (resultado == 2) então
                imprima "paquerador(a)"
        fim-se
        se (resultado == 3) então
                imprima "atraente"
        fim-se
        se (resultado == 4) então
                imprima "irresistível"
        fim-se
fim-algoritmo
```

```c
int main() {
	// declare as variaveis
	int ano, v1, v2, resultado;

	// leia o ano
	printf("Insira seu ano de nascimento: ");
	scanf("%i", &ano);
	v1 = ano/100;
	v2 = ano - v1*100;

	// Formar o resulado
	resultado = (v1+v2)%5;
	if (resultado == 0)
		printf("\nVoce e timido");
	if (resultado == 1)
		printf("\nVoce e sonhador");
	if (resultado == 2)
		printf("\nVoce e paquerador");
	if (resultado == 3)
		printf("\nVoce e atraente");
	if (resultado == 4)
		printf("\nVoce e irresistivel");
}
```

11 - Dado um número real não negativo, informe sua raiz quadrada.

{ Através do método de aproximações sucessivas, chamado Método Babilônico
Pede-se o numero do qual quer se extrair a raíz, uma aproximação (que pode ser qualquer número aleatório), e o erro
Sendo S o numero de que ser quer extrair a raiz e x a aproximação, usa-se a **for**mula ½(x+S/x), e ver**if**ica-se elevando a resposta ao quadrado. Caso o erro seja maior que o estipulado, usa-se o valor obtido como uma nova aproximação }

algoritmo
declare raiz, aproximacao, erro (numerico)
leia raiz
leia aproximacao
leia erro
enquanto (absoluto((aproximacao*aproximacao) – raiz) > erro) **faça**
aproximação <- ((aproximacao*(raiz/aproximacao))/2)
fim-enquanto
imprima aproximacao
fim-algoritmo

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

int main() {
	float raiz, aproximacao, erro;
	printf("Insira o numero do qual se quer tirar a raiz quadrada: ");
	scanf("%f", &raiz);
	printf("Insira uma aproximacao: ");
	scanf("%f", &aproximacao);
	printf("Insira o erro: ");
	scanf("%f", &erro);
	while (abs((aproximacao*aproximacao)-raiz) > erro)
		aproximacao = ((aproximacao+(raiz/aproximacao))/2);
	printf("\nA raiz quadrada aproximada de %f e %f", raiz, aproximacao);
}
```