Nome: Luiz Felipe _____

1. O departamento de trânsito do estado anotou dados de acidentes de trânsito no último ano. Para cada motorista envolvido no acidente, tem-se a seguinte informação:
- Ano de nascimento
- Sexo (M-masculino, F-feminino)
- Procedência (0-Capital, 1-Interior, 2-Outro Estado)

Faça um algoritmo que:

A) Calcule quantos motoristas tem menos de 21 anos;

Resposta:

```
import java.util.*;


class Motorista
{
	Date anoNasc;
	char sexo;
	int procedencia;

	public Motorista(Date ano, char s, int p)throws Exception
	{
		if( (p < 0) || (p>2)) throw new Exception ("Procedencia invalida");

		if("MF".indexOf(Character.toUpperCase(s)) == -1)throw new Exception ("Sexo
invalido");


		this.anoNasc = ano;
		this.sexo = Character.toUpperCase(s);
		this.procedencia = p;
	}

	public Date getNasc()
	{
		return this.anoNasc;
	}
	public char getSexo ()
	{
		return this.sexo;
	}
	public int getProcedencia()
	{
		return this.procedencia;
	}
}

class Historico
{
		Vector<Motorista> hist;

		public Historico()
		{
			this.hist = new Vector<Motorista>();
```

```
	}

	public void addMot(Motorista m)
	{
		hist.add(m);
	}

	public int menos21()
	{
		int qts =0;

		for(int i = 0; i<this.hist.size();i++)
		{
			Motorista temp = this.hist.get(i);
			Date ref = new Date();
			ref.setYear(ref.getYear()-21);
			boolean status = temp.getNasc().after(ref);
			if(status){qts++;}

		}

		return qts;
	}
}

public class MainAcidentes {

	public static void main (String[] args) {
		try{
			Motorista mt = new Motorista(new Date(97,1,12),'M',1);
			Historico hs = new Historico();
			hs.addMot(mt);

			mt = new Motorista(new Date(101,1,12),'M',1);
			hs.addMot(mt);

			mt = new Motorista(new Date(87,1,12),'M',1);
			hs.addMot(mt);

			mt = new Motorista(new Date(78,1,12),'M',1);
			hs.addMot(mt);

			mt = new Motorista(new Date(105,1,12),'M',1);
			hs.addMot(mt);

			System.out.println(hs.menos21());
			}catch(Exception erro)
			{
				System.err.println(erro.getMessage());
			}
	}
}
```

B) Quantas mulheres são da capital e quantas são do interior;

Resposta:

```
import java.util.*;


class Motorista
{
	Date anoNasc;
	char sexo;
	int procedencia;

	public Motorista(Date ano, char s, int p)throws Exception
	{
		if( (p < 0) || (p>2)) throw new Exception ("Procedencia invalida");

		if("MF".indexOf(Character.toUpperCase(s)) == -1)throw new Exception ("Sexo invalido");


		this.anoNasc = ano;
		this.sexo = Character.toUpperCase(s);
		this.procedencia = p;
	}

	public Date getNasc()
	{
		return this.anoNasc;
	}
	public char getSexo ()
	{
		return this.sexo;
	}
	public int getProcedencia()
	{
		return this.procedencia;
	}
}

class Historico
{
		Vector<Motorista> hist;

		public Historico()
		{
			this.hist = new Vector<Motorista>();
		}
```

```
	public void addMot(Motorista m)
	{
		hist.add(m);
	}

	public int menos21()
	{
		int qts =0;

		for(int i = 0; i<this.hist.size();i++)
		{
			Motorista temp = this.hist.get(i);
			Date ref = new Date();
			ref.setYear(ref.getYear()-21);
			boolean status = temp.getNasc().after(ref);
			if(status){qts++;}

		}

		return qts;
	}

	public int qntsPessoas (char s, int p) throws Exception
	{
		if( (p < 0) || (p>2)) throw new Exception ("Procedencia invalida");

		if("MF".indexOf(Character.toUpperCase(s)) == -1)throw new Exception ("Sexo invalido");

		int qts = 0;

		for(int i = 0; i<this.hist.size();i++)
		{
			Motorista temp = this.hist.get(i);
			if( (temp.getSexo() ==Character.toUpperCase(s)) && (temp.getProcedencia()== p) )
			{
				qts++;
			}

		}


		return qts;
	}
}

public class MainAcidentes {

	public static void main (String[] args) {
		try{
			Motorista mt = new Motorista(new Date(97,1,12),'M',1);
			Historico hs = new Historico();
			hs.addMot(mt);
```

```
                mt = new Motorista(new Date(101,1,12),'M',1);
                hs.addMot(mt);

                mt = new Motorista(new Date(87,1,12),'f',1);
                hs.addMot(mt);

                mt = new Motorista(new Date(78,1,12),'f',1);
                hs.addMot(mt);

                mt = new Motorista(new Date(105,1,12),'M',1);
                hs.addMot(mt);

                mt = new Motorista(new Date(105,1,12),'F',0);
                hs.addMot(mt);

                System.out.println("Qtd mulheres da capital: "+hs.qntsPessoas('F',0));
                System.out.println("Qtd mulheres do interior: "+hs.qntsPessoas('F',1));
                }catch(Exception erro)
                {
                        System.err.println(erro.getMessage());
                }
        }
}
```

C) Calcule quantos motoristas do interior do estado tem idade maior que 60 anos;

Resposta:

```
import java.util.*;


class Motorista
{
        Date anoNasc;
        char sexo;
        int procedencia;

        public Motorista(Date ano, char s, int p)throws Exception
        {
                if( (p < 0) || (p>2)) throw new Exception ("Procedencia invalida");

                if("MF".indexOf(Character.toUpperCase(s)) == -1)throw new Exception ("Sexo
invalido");


                this.anoNasc = ano;
                this.sexo = Character.toUpperCase(s);
                this.procedencia = p;
        }

        public Date getNasc()
        {
                return this.anoNasc;
        }
        public char getSexo ()
```

```
	{
		return this.sexo;
	}
	public int getProcedencia()
	{
		return this.procedencia;
	}
}

class Historico
{
		Vector<Motorista> hist;

		public Historico()
		{
			this.hist = new Vector<Motorista>();
		}

		public void addMot(Motorista m)
		{
			hist.add(m);
		}

		public int menos21()
		{
			int qts =0;

			for(int i = 0; i<this.hist.size();i++)
			{
				Motorista temp = this.hist.get(i);
				Date ref = new Date();
				ref.setYear(ref.getYear()-21);
				boolean status = temp.getNasc().after(ref);
				if(status){qts++;}

			}

			return qts;
		}

		public int qntsPessoas (char s, int p) throws Exception
		{
			if( (p < 0) || (p>2)) throw new Exception ("Procedencia invalida");

			if("MF".indexOf(Character.toUpperCase(s)) == -1)throw new Exception ("Sexo
invalido");

			int qts = 0;

			for(int i = 0; i<this.hist.size();i++)
			{
				Motorista temp = this.hist.get(i);
				if( (temp.getSexo() ==Character.toUpperCase(s)) &&
(temp.getProcedencia()== p) )
				{
					qts++;
```

```
                        }

                }


                return qts;
        }

        public int mais60 (int p) throws Exception
        {
                if( (p < 0) || (p>2)) throw new Exception ("Procedencia invalida");

                int qts =0;

                for(int i = 0; i<this.hist.size();i++)
                {
                        Motorista temp = this.hist.get(i);
                        if(temp.getProcedencia()==p){
                                Date ref = new Date();
                                ref.setYear(ref.getYear()-60);
                                boolean status = temp.getNasc().before(ref);
                                if(status){qts++;}
                                }

                }

                return qts;
        }
}

public class MainAcidentes {

        public static void main (String[] args) {
                try{
                        Motorista mt = new Motorista(new Date(97,1,12),'M',1);
                        Historico hs = new Historico();
                        hs.addMot(mt);

                        mt = new Motorista(new Date(101,1,12),'M',1);
                        hs.addMot(mt);

                        mt = new Motorista(new Date(87,1,12),'f',1);
                        hs.addMot(mt);

                        mt = new Motorista(new Date(78,1,12),'f',1);
                        hs.addMot(mt);

                        mt = new Motorista(new Date(105,1,12),'M',1);
                        hs.addMot(mt);

                        mt = new Motorista(new Date(105,1,12),'F',0);
                        hs.addMot(mt);
                        mt = new Motorista(new Date(32,1,12),'F',1);
                        hs.addMot(mt);

                        System.out.println("Qtd pessoas com mais de 60 anos: "+hs.mais60(1));
```

```
                    }catch(Exception erro)
                    {
                            System.err.println(erro.getMessage());
                    }
        }
}
```

2. O algoritmo abaixo tem o objetivo de encontrar o maior entre n números. A condição de parada é a entrada de um valor 0, ou seja, o algoritmo deve ficar calculando o maior até que a entrada seja igual a zero.
Existe um erro que faz com que este algoritmo não dê o resultado correto. Encontre o erro e dê a solução para que o maior número seja realmente encontrado.

```
var
N,MAIOR: INTEIRO

inicio


   ESCREVAL ("DIGITE UM NUMERO")
   LEIA (N)
   MAIOR ✇ N

ENQUANTO (N<>0) FACA

   ESCREVAL ("DIGITE UM NUMERO")
   LEIA (N)

   SE (N > MAIOR) ENTAO
         N ✇ MAIOR
   FIMSE

FIMENQUANTO

ESCREVAL ("O NUMERO NUMERO DIGITADO FOI ", MAIOR)

fimalgoritmo
```

Resposta:
O erro está na atribuição (N <- MAIOR) que está dentro do **IF(SE)**, o correto seria o atribuir o valor da variável N á variável MAIOR, ou seja, inverter a atribuição para (MAIOR <- N).

3. Crie um algoritmo que determine se um número é primo ou não.

Definição Número Primo: Um número é primo quando é inteiro positivo, distinto de 0 e 1 e que seus únicos divisores positivos forem 1 e ele próprio. Por exemplo: O número 13 é primo pois ele é diferente de 0, é diferente de 1 e o resultado da divisão só é exato quando ele é dividido por 1 e 13.

Resposta: (Utilizando Linguagem Java)

```
import java.io.*;

class Primo{

	public static void main(String args[]){


		BufferedReader teclado =  new BufferedReader (new InputStreamReader (
	System.in));
		boolean invalido = false;
		int valor=0;

		do{
			try{
				System.out.println(“Digite um numero inteiro”);
				valor = Integer.parseInt(teclado.readLine());
				invalido = false;

			   }catch(IOException erro)
			{
				invalido = true;
				System.err.println(erro.getMessage());
			}
			catch (NumberFormatException erro)
			{
				invalido = true;
				System.err.println(“Input invalid, por favor digite um numero inteiro”);
			}


		}while(invalido);

		if(valor < 2){
			System.out.println(“O numero: ”+valor+”, nao e um numero primo.”);
		}else{
			int div = 0;
			for (int i = 1; i<= valor;i++ )
			{
				If(valor % i == 0){
				div++;
				}
			}
			If(div == 2){
				System.out.println(“O numero: ”+valor+”, e um numero primo.”);
			}else{
				System.out.println(“O numero: ”+valor+”, nao e um numero primo.”);
			}
		}

	}
}
```

4. Joãozinho foi ao correio postar uma carta para sua namorada Mariazinha. Ao ver a funcionária do correio entupir sua carta de selos, Joãozinho chegou em casa empenhado em resolver esse problema, pensou em criar um algoritmo para calcular o menor número de selos necessários de acordo com o valor da postagem. Joãozinho sabe que o valor da postagem sempre será maior ou igual a 8 centavos e que existem selos de 1, 3 e 5 centavos.

Exemplo: Para uma postagem no valor de 18 centavos. Teríamos 3 selos de 5 centavos e 1 selo de 3 centavos. O valor somado de todos os selos deve ser igual ao valor da postagem (sem troco).
Como Joãozinho faltou na aula de algoritmos ajude-o a resolver esse problema

Resposta: (Código em Java).

```
class Selos
{

	public static int[] getSelosMin (int valorCentavos) throws Exception
	{
		if(valorCentavos <8) throw new Exception ("valor invalido, deve ser um valor maior ou igual a 8 centavos");

		int qtdSelo5 = 0;
		int qtdSelo3 = 0;
		int qtdSelo1 = 0;

		//X.selo5 + Y.selo3 + Z.selo1;
		while(valorCentavos >= 5)
		{
			valorCentavos = valorCentavos - 5;
			qtdSelo5++;

		}

		while(valorCentavos >= 3)
		{
			valorCentavos = valorCentavos - 3;
			qtdSelo3++;

		}

		while(valorCentavos >= 1)
		{
			valorCentavos = valorCentavos - 1;
			qtdSelo1++;

		}


		int[] temp = new int[3];
		temp[0] = qtdSelo5;
		temp[1] = qtdSelo3;
		temp[2] = qtdSelo1;

		return temp;
	}
}
```

```
public class Postagem {

	public static void main (String[] args) {
		try{
			int[] temp = Selos.getSelosMin(70); //Valor de exemplo
			System.out.println("Quantidade de selos de 5 centavos: "+temp[0]);
			System.out.println("Quantidade de selos de 3 centavos: "+temp[1]);
			System.out.println("Quantidade de selos de 1 centavo: "+temp[2]);
			}catch(Exception e)
			{
				System.err.println(e.getMessage());
			}

	}
}
```

5. Dado o trecho de código abaixo:

```
class Veiculo {
        protected void acelerar() {
                System.out.println("Veículo acelerando...");
        }
}

public class Carro extends Veiculo {
        protected final void acelerar() {
                System.out.println("Carro acelerando...");
        }
        public static void main(String [] args) {
                new Carro().acelerar();
        }
}
```

Qual o resultado do código acima?

A) Carro acelerando...
B) Veículo acelerando...
C) Veículo acelerando ...
   Carro acelerando...
D) Carro acelerando...
   Veículo acelerando...
E) O código não compila

Resposta: E

6. Dado o trecho de código abaixo:

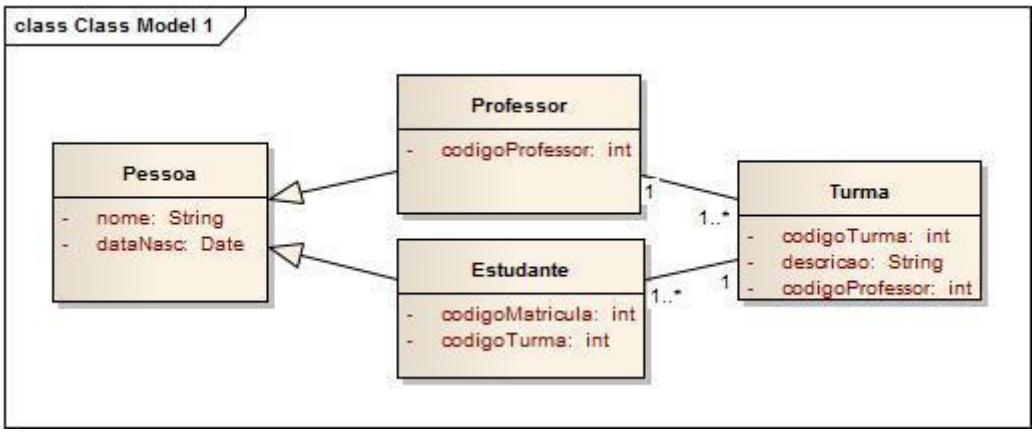
```
class Chevrolet {}
class Monza extends Chevrolet {}

class Fabrica {
        public static void main(String [] args) {
                Monza m1 = new Monza();
                Chevrolet chev1 = new Chevrolet();
                Chevrolet chev2 = m1;
                //insira seu código aqui
        }
}
```

Qual letra abaixo, inserida na linha 9, compilará? (Escolha todas que se aplicam)

A) Monza m2 = (Monza) chev1;
B) Monza m3 = (Monza) chev2;
C) Monza m4 = chev2;
D) Nenhuma das anteriores compilará.

Resposta: B(perfeito) e A(compilará porém dará erro)

7. De acordo com o diagrama abaixo, escreva as classes em Java correspondentes.



Resposta:

class Turma{

private String descricao;
private int codigoTurma;
private int codigoProfessor;

```
public Turma(String desc,int cdTurma, int cdProf)
{
        this.descricao = desc;
        this.codigoTurma = cdTurma;
        this.codigoProfessor = cdProf;
}

public void setDesc (String desc)
{
        this.descricao = desc;
}

public void setcdTurma (int cdTurma)
{
        this.codigoTurma = cdTurma;
}

public void setcdProf (int cdProf)
{
        this.codigoProfessor = cdProf;
}

public String getDesc()
{
        return this.descricao;
}

public int getcdTurma()
{
        return this.codigoTurma;
}
```

```
	public int getcdProf()
	{
		return this.codigoProfessor;
	}


	}



import java.util.Date;

public class Pessoa {

	protected String nome;
	protected  Date dataNasc;

	public Pessoa(String nm, Date data)
	{
		this.nome = nm;
		this.dataNasc = data;
	}

	public String getNome()
	{
		return this.nome;
	}

	public Date getDataNasc()
	{
		return this.dataNasc;
	}

	public void setNome(String nm)
	{
		this.nome = nm;
	}

	public void setDataNasc (Date data)
	{
		this.dataNasc = data;
	}

}


import java.util.Date;
public class Professor extends Pessoa {
	private int codigoProfessor;

	public Professor(int cdProf)
	{
		super("",new Date());
		this.codigoProfessor = cdProf;
	}
```

```
	public Professor(String nome , Date data, int cdProf)
	{
		super(nome,data);
		this.codigoProfessor = cdProf;
	}

	public void setcdProf(int cdProf)
	{
		this.codigoProfessor = cdProf;
	}

	public int getcdProf()
	{
		return this.codigoProfessor;
	}

}



import java.util.Date;
public class Estudante extends Pessoa {
	int codigoMatricula;
	int codigoTurma;

	public Estudante(int Turma, int Mat)
	{
		super("",new Date());
		this.codigoTurma = Turma;
		this.codigoMatricula = Mat;
	}
	public Estudante(String nome, Date data, int Turma, int Mat)
	{
		super(nome,data);
		this.codigoTurma = Turma;
		this.codigoMatricula = Mat;
	}

	public void setcdMat(int Mat)
	{
		this.codigoMatricula = Mat;
	}

	public void setcdTurma (int Turma)
	{
		this.codigoTurma = Turma;
	}

	public int getMat()
	{
		return this.codigoMatricula;
	}

	public int getTurma()
	{
		return this.codigoTurma;
```

```
        }

}
```

8. Suponha que a classe Pessoa do exercício anterior possua o seguinte método:

void andar() { System.out.println("Pessoa andando"); }

E que a classe Estudante tenha os seguintes métodos:

void andar() { System.out.println("Estudante andando"); }
void andar(String percurso) { System.out.println("Percurso="+percurso); }

Dê a saída de cada código abaixo e **justifique**:

a) Pessoa p = new Pessoa(); p.andar();

Se for utilizado o exemplo exato do item anterior aconteceria um erro de compilação, pois minha classe Pessoa não possui o construtor sem parâmetros, porém caso fosse criado um construtor sem parâmetro ou fosse passados os parâmetros corretos**, o método andar() escreveria na tela do usuário "Pessoa andando".**

b) Estudante e = new Estudante(); e.andar();

Se for utilizado o exemplo exato do item anterior aconteceria um erro de compilação, pois minhas classes Pessoa e Estudante não possuem construtores sem parâmetros, porém caso fosse criado os respectivos construtores sem parâmetros ou fossem passados os parâmetros corretos, **o método andar() de Estudante escreveria na tela do usuário "Estudante andando", pois a classe Estudante reescreveu o método andar() da classe pai (superclasse), ou seja, aconteceu polimorfismo.**

c) Pessoa h = new Estudante(); h.andar();

O objeto h seria uma instância da classe Estudante, portanto ao chamar h.andar(), seria executado o método andar da classe Estudante, ou seja, escreveria na tela do usuário "Estudante andando", pois a classe Estudante reescreveu o método andar() da classe pai(superclasse), ou seja, aconteceu polimorfismo.

d) Estudante e1 = new Estudante(); e1.andar("200");

O método andar(String) de Estudante, seria executando, escrevendo na tela do usuário "Percurso= 200", pois a classe Estudante possui sobrecarga do método andar, portanto o método adequado seria escolhido utilizando a assinatura do método.

e) Pessoa h1 = new Pessoa(); h1.andar("200");

Aconteceria um erro de compilação, pois a classe pai Pessoa, não possui um método com a assinatura andar(String), a classe que possui tal método é a classe Estudante.

f) Pessoa p2 = new Estudante(); p2.andar("200");

Aconteceria um erro de compilação, pois a classe pai Pessoa, não possui um método com a assinatura andar(String), a classe que possui tal método é a classe Estudante. Seria necessário alterar o comando acima para: Estudante p2 = new Estudante; p2.andar("200");

9. O algoritmo abaixo lê 3 notas de um aluno, calcula a média e informa se ele foi aprovado, ficou de recuperação ou foi reprovado. A média de aprovação é >= 7.0; a média de recuperação é >= 5.0 e < 7.0; e a média do reprovado é < 5.0

```
import java.util.Scanner;

public class Principal {

public static void main(String[] args) {

   Scanner ler = new Scanner(System.in);
   double nota1, nota2, nota3, media;

   System.out.println("Entre com a primeira nota:");
   nota1 = ler.nextDouble();
   System.out.println("Entre com a segunda nota:");
   nota2 = ler.nextDouble();
   System.out.println("Entre com a terceira nota:");
   nota3 = ler.nextDouble();

   media = (nota1+nota2+nota3)/3;

   if(media >= 7) {
      System.out.println("Aprovado com média "+media);
   }
   else if(media >=5) {
      System.out.println("Recuperação com média "+media);
   }
   else {
      System.out.println("Reprovado com média "+media);
   }
}
}
```

Com base no algoritmo acima, responda as seguintes questões.

A) O que acontece se o usuário digitar uma letra no lugar de um número?

Resposta:

A instância de objeto “ler” da classe Scanner tenta realizar uma conversão para double com o input do usuário, no caso, como o input é uma letra não será possível realizar tal conversão, portanto é gerado um erro de formato, e como tal erro não é tratado com um “try{}catch(){}”, o erro “estoura” na tela do usuário e o programa acaba.

B) Altere o código para evitar que a situação descrita na questão anterior ocorra.

Resposta:

```
import java.util.*;

public class Principal {

public static void main(String[] args) {

double nota1 = 0, nota2 = 0, nota3 = 0, media;
boolean invalido = false;
do{
    try{

            Scanner ler = new Scanner(System.in);

            System.out.println("Entre com a primeira nota:");
            nota1 = ler.nextDouble();
            System.out.println("Entre com a segunda nota:");
            nota2 = ler.nextDouble();
            System.out.println("Entre com a terceira nota:");
            nota3 = ler.nextDouble();
            invalido = false;

            }catch(InputMismatchException  erro)
            {
                    System.err.println("Input de formato invalido, por favor digite um numero");
                    invalido = true;

            }
            catch(Exception erro)
            {
                    System.err.println("Erro: "+erro.getMessage());
                    invalido = true;

            }
}while(invalido);

media = (nota1+nota2+nota3)/3;

if(media >= 7) {
System.out.println("Aprovado com média "+media);
}
else if(media >=5) {
System.out.println("Recuperação com média "+media);
}
else {
System.out.println("Reprovado com média "+media);
}
}
}
```

C) Altere o código para que o programa leia as notas de uma turma de 10 alunos e armazene os resultados em uma coleção. Após o cálculo das médias de todos os alunos da turma o programa deverá percorrer esta coleção, calcular e exibir a média geral da turma.

Resposta:

```
import java.util.*;

public class Principal {

public static void main(String[] args) {



double nota1 = 0, nota2 = 0, nota3 = 0, media;
int qtsAlunos = 1;
Vector<Double> meuVetor = new Vector<Double>(30);
do{
    try{

        Scanner ler = new Scanner(System.in);

        System.out.println("Aluno: "+qtsAlunos+", Entre com a primeira nota:");
        nota1 = ler.nextDouble();
        System.out.println("Aluno: "+qtsAlunos+", Entre com a segunda nota:");
        nota2 = ler.nextDouble();
        System.out.println("Aluno: "+qtsAlunos+", Entre com a terceira nota:");
        nota3 = ler.nextDouble();


        media = (nota1+nota2+nota3)/3;


        if(media >= 7) {
        System.out.println("Aprovado com media "+media);
        }
        else if(media >=5) {
        System.out.println("Recuperacao com media "+media);
        }
        else {
        System.out.println("Reprovado com media "+media);
        }

        meuVetor.add(nota1);
        meuVetor.add(nota2);
        meuVetor.add(nota3);

        qtsAlunos++;

        }catch(InputMismatchException  erro)
        {
                System.err.println("Input de formato invalido, por favor digite um numero");
```

```
        }
        catch(Exception erro)
        {
                System.err.println("Erro: "+erro.getMessage());

        }
}while(qtsAlunos <= 10);


double mediaFinal = 0;

for(int i = 0 ; i<meuVetor.size();i++)
{
    mediaFinal = mediaFinal+ meuVetor.get(i);
}


mediaFinal = mediaFinal/meuVetor.size();

System.out.println("Media geral de toda a classe: "+mediaFinal);



}
}
```

10. Considere um banco de dados relacional formado pelas tabelas DISCIPLINAS, PROFESSORES e ALUNOS, com conteúdos definidos conforme o esquema a seguir:

DISCIPLINAS

CodDisciplina	Nome	Professor	Aluno	NotaFinal

PROFESSORES

CodProfessor	Nome	Endereco	Sala	Telefone

ALUNOS

CodAluno	Nome	Endereco	DataIngresso

Dada a seguinte consulta SQL:

```
SELECT DISCIPLINAS.Nome, PROFESSORES.Nome, ALUNOS.Nome
FROM DISCIPLINAS, PROFESSORES, ALUNOS
WHERE ( DISCIPLINA.Professor = PROFESSORES.CodProfessor and
        DISCIPLINA.Aluno = ALUNOS.CodAluno and
        DISCIPLINA.NotaFinal = 0 )
```

Assinale a alternativa que apresenta corretamente o resultado dessa consulta.

a) A lista dos alunos com freqüência ZERO em algum curso ministrado por um determinado professor.
b) A lista das disciplinas em que houve alunos com nota ZERO. Essa lista é uma tabela com três colunas, sendo a primeira coluna o nome da disciplina e as segunda e terceira colunas os nomes do professor que ministrou a disciplina e do aluno que tirou nota ZERO, respectivamente.
c) A lista das disciplinas em que um determinado aluno tirou nota ZERO com algum professor.
d) A lista dos códigos de disciplina, código de aluno e código de professor, tal que cada aluno tirou uma nota próxima de ZERO na disciplina.
e) A junção das três tabelas.

Resposta: B

11. Dado um banco de dados relacional formado pela tabela abaixo:

PROJETO

Cód_Projeto	Verba	Depto	Cod_Gerente
10000	200000	10	1001
11000	150000	30	1002
12000	300000	20	1001
13000	249000	20	1002
14000	112000	40	1003
15000	123000	60	1004
16500	45000	60	1005
16900	345000	10	1005
17000	22000	10	1004

O comando SQL que lista todos os projetos do departamento 10 que têm verba superior a 100000 é:

a) SELECT PROJETO WITH depto 10 AND verba > 100000
b) SELECT cod_projeto WITH depto = 10, verba > 100000
c) SELECT * FROM PROJETO WHERE depto = 10 AND verba > 100000

d) SELECT cod_projeto BETWEEN depto = 10 , verba > 100000
e) SELECT * PROJETO WITH depto = 10 ^ verba > 100000

Resposta: C

12. Considere os seguintes esquemas relacionais:

Fornecedores(id_f, nome_f, cidade, estado)
Peças(id_p, nome_p, cor, preco)
Catálogo(id_f, id_p)

a) Elabore uma consulta SQL para obter as cidades que possuem fornecedores no estado de São Paulo:

Resposta:

SELECT F.cidade **FROM** Fornecedores as F
WHERE F.estado = 'São Paulo'

b) Elabore uma consulta SQL para obter o total de quantidade de peças cujo preço é maior do que 10:

Resposta:

SELECT COUNT(*) **FROM** Peças as P **WHERE** P.preco > 10

c) Elabore uma consulta SQL para obter as cores disponíveis da peça cujo nome é "Parafusos" do fornecedor cujo nome é "Casa do Parafuso":

Resposta:

SELECT P.cor **FROM** Peças as P, Fornecedor as F, Catalogo as C
WHERE F.id_f = C.id_F AND P.id_p = C.id_P AND P.nome_p = 'Parafusos' AND F.nome_f = 'Casa do Parafuso'