Obs: O arquivo lab1.zip contém o código por completo. Abaixo estão somente os códigos necessários para o relatório.

Item 05)

Alterando a função imprimir da classe Documento para o código abaixo, obtém-se o desejado:

```
  public void imprimir( ) {
    System.out.println( "autor: " + this.autor + "\ndataDeChegada: " + this.dataDeChegada);
  }
```

Código de ExemploDocumento.java:

```
package br.unicamp.ic.mc302.documento;
class ExemploDocumento{
  static public void main(String args[ ]){
    Documento d1; // declaração de uma referência para um
            // objeto do tipo Documento
    d1 = new Documento( ); //alocação dinâmica de memória para a
                    //criação do objeto
    d1.criarDocumento("Camila",181101);
    d1.imprimir( ); // envio de mensagem para o objeto
    d1.editar();
    Documento d2;
    d2 = new Documento();
    d2.criarDocumento("Bob",18181);
    d2.imprimir();
  }
}
```

Item 07)

Ambos métodos creditam, no caso de alterar diretamente na variável, deve-se tomar o cuidado de fazer a soma.

```
c1.saldoAtual = c1.saldoAtual + 100
```

Item 08)

Utilizando a função abaixo, transfere-se do objeto que recebe a chamada de transferir, para a conta passada como parâmetro.

```
  public void transferir(ContaCor conta, double quantia, int senha){
        if(this.senha == senha){
            if(this.saldoAtual >= quantia){
                this.debitaValor(quantia, senha);
                conta.creditaValor(quantia);
            }
        }
  }
```

Item 09)

Código de Exemplo.java abaixo:

```
package br.unicamp.ic.mc302.contaCor;
class Exemplo{
   static public void main(String args[ ]){
      ContaCor c1 = new ContaCor("Ursula", 500, 1, 1);
      c1.creditaValor(100);
      c1.debitaValor(50 ,1);
      c1.saldoAtual += 100;
      ContaCor c2 = new ContaCor("Bob", 500, 2, 2);
      c1.transferir(c2, 100, 1);
      System.out.println("Saldo final c1 = " + c1.getSaldo(1));
      System.out.println("Saldo final c2 = " + c2.getSaldo(2));
   }
}
```

Item 10)

O construtor de MeuVetor tem como parâmetro o tamanho do vetor a ser usado. O método setValor(pos, valor), seta a posição do vetor "pos" com "valor". O método intercala retorna o novo MeuVetor com os dois MeuVetor intercalados.

Código de MeuVetor.java:

```
package br.unicamp.ic.mc302.meuVetor;
public class MeuVetor {
	private int vetor[];
	private int tamanho;
	public MeuVetor(int tamanho){
		this.vetor = new int[tamanho];
		this.tamanho = tamanho;
	}
	public int getTamanho(){
		return this.tamanho;
	}
	public void setValor(int pos, int valor){
		this.vetor[pos] = valor;
	}
	public int getValor(int pos){
		return this.vetor[pos];
	}
	public MeuVetor intercala(MeuVetor vetor2){
		MeuVetor v = new MeuVetor(this.getTamanho() + vetor2.getTamanho());
		int i = 0;
		while(i < this.getTamanho() || i < vetor2.getTamanho()){
			if(i < this.getTamanho()){
				v.setValor(i*2, this.getValor(i));
			}
			if (i < vetor2.getTamanho()){
				v.setValor(i*2+1, vetor2.getValor(i));
			}
			i++;
		}
		v.print();
		return v;
	}
```

```
	public void print(){
		for (int i = 0; i < tamanho; i++){
			System.out.println(i + ": " + this.vetor[i]);
		}
	}
}
```

Código de ExemploMeuVetor.java:

```
package br.unicamp.ic.mc302.meuVetor;
public class ExemploMeuVetor {
	static public void main(String args[ ]){
		MeuVetor v1 = new MeuVetor(4);
		MeuVetor v2 =  new MeuVetor(4);
		v1.setValor(0, 0);
		v1.setValor(1, 2);
		v1.setValor(2, 4);
		v1.setValor(3, 6);
		v2.setValor(0, 1);
		v2.setValor(1, 3);
		v2.setValor(2, 5);
		v2.setValor(3, 7);
		MeuVetor v3 = v1.intercala(v2);
		v3.print();
	}
}
```