FATEC
Faculdade de Tecnologia do Estado de São Paulo

Laboratório de Engenharia de Software

Atividade para N2

Projeto e Implementação Orientado a Objetos

Regras:

- Deve ser realizado na mesma equipe de projeto da disciplina;
- Entrega até 22/10/15;
- Enviar o projeto Java e arquivo PDF com a documentação em arquivo compactado para marco.furlandesouza@fatec.sp.gov.br;
- Vale até 5,0 pontos na nota N2.

O problema

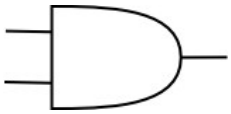
Deseja-se criar um **simulador simples** de **circuitos lógicos**, cujos requisitos são:

Requisito	Descrição
R001	O sistema deverá permitir a criação de **portas lógicas**: elementos simples que executarão operações lógicas tais como AND, OR, NOT, XOR etc.
R002	As portas lógicas deverão ter de 1 a 2 entradas e possuir apenas uma única saída que é a aplicação da sua função às entradas.
R003	O sistema deverá permitir a criação de **subsistemas lógicos**: um conjunto internamente conectado de portas lógicas realizando uma operação que é consequência dessas conexões.
R004	O número mínimo de entradas e de saídas dos subsistemas lógicos é 1 e o número máximo é arbitrário.
R005	O sistema deverá permitir a simulação dos circuitos por meio de um programa de testes que implemente tabelas verdade para os circuitos definidos.
R006	O sistema deverá ser implementado com uma interface em linha de comando, apresentando os testes de modo organizado.

Exemplos sugeridos

Portas lógicas

Exemplo de porta AND

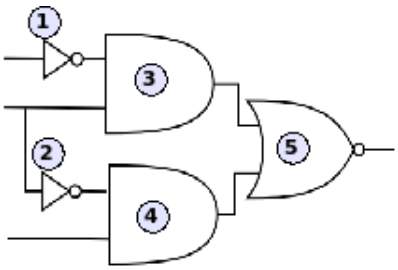


Uma sugestão de abstração:

```
portaAND  = new And(2,1); //2 entradas e 1 saída
portaAND.Entrada(1).Nivel = true; //entrada 1 alta
portaAND.Entrada(2).Nivel = true; //entrada 2 alta
bool valor = portaAND.Saida(1).Nivel; //valor vale true
...
```

Subsistema lógico

Neste caso, (1) e (2) são portas NOT, (3) e (4) são portas AND e (5) é uma porta NOR:



Uma sugestão de abstração:

```
//exemplo de criação do circuito
circuito = new Circuito(3,1); //3 entradas e 1 saída
//elemento 1
circuito.Adicionar("INV1", new Inversor(1,1));
//elemento 2
circuito.Adicionar("INV2", new Inversor(1,1));
//elemento 3
circuito.Adicionar("AND1", new And(2,1));
//elemento 4
circuito.Adicionar("AND2", new And(2,1));
//elemento 5
circuito.Adicionar("NOR1", new Nor(2,1));
//conexões
circuito.Entrada(1).Conectar(
        circuito.Componente("INV1").Entrada(1));
circuito.Entrada(2).Conectar(
        circuito.Componente("AND1").Entrada(2));
circuito.Entrada(2).Conectar(
        circuito.Componente("INV2").Entrada(1));
circuito.Entrada(3).Conectar(
        circuito.Componente("AND2").Entrada(2));
```

```
        circuito.Saida(1).Conectar(
                circuito.Componente("NOR1").Saida(1));
        circuito.Componente("INV1").Saida(1).Conectar(
                circuito.Componente("AND1").Entrada(1));
        circuito.Componente("INV2").Saida(1).Conectar(
                circuito.Componente("AND2").Entrada(1));
        circuito.Componente("AND1").Saida(1).Conectar(
                circuito.Componente("NOR1").Entrada(1));
        circuito.Componente("AND2").Saida(1).Conectar(
                circuito.Componente("NOR1").Entrada(2));
        //níveis das entradas
        circuito.Entrada(1).setNivel(true);
        circuito.Entrada(2).setNivel(false);
        circuito.Entrada(3).setNivel(true);
        //saída - a função recalcula o circuito
        bool valor = circuito.Saida(1).getNivel();
        //…depois pode-se exibir os valores - a cargo do programador
```

O que é para fazer?

1. Procurar padrões de projeto que poderiam ser utilizados na resolução deste problema: indicar o nome, referência e justificar sua escolha.
2. Elaborar um diagrama UML de classes do projeto do sistema e um diagrama de sequências UML que exemplifique um cenário de utilização do sistema (por exemplo, a construção e uso do segundo circuito de exemplo desta lista).
3. Implementar o sistema em Java. Não é necessário criar uma interface gráfica – o sistema pode funcionar em linha de comando, se preferir.