```
                doIt();

        }

}

//Read value and see if it exists
public boolean read(short address) {
        short tag = (short) ( (address & 0xF00) >>> 8);
        short slot = (short) ( (address & 0x0F0) >>> 4);
        short offset = (short) (address & 0x00F);

        if( tag == this.node[slot].getTag() ) {

                if( this.node[slot].getValid() ) {
                        System.out.printf( "At that byte there is the value  %x (Cache Hit)\n",
                                        this.node[slot].getData(offset) );
                        return true;
                }

                else {

                        this.data(address);
                        System.out.printf( "At that byte there is the value %x (Cache Miss)\n",
                                        this.node[slot].getData(offset) );
                        return true;
                }
        } else {

                if( !this.node[slot].getValid() ) {

                        this.data(address);
                        System.out.printf( "At that byte there is the value %x (Cache Miss)\n",
                                        this.node[slot].getData(offset) );
                        return true;

                } else {

                        this.data(address);
                        System.out.printf( "At that byte there is the value %x  (Cache Miss)\n",
                                        this.node[slot].getData(offset) );
                        return true;
                }
        }
}

//Initialize Cache
public void initializeCache() {
        for (short s = 0; s < MAIN_MEMORY; s++) {
                this.memory[s] = (short) (s & 0Xff);
        }

        for (byte b = 0; b < SIZE; b++) {
                this.node[b] = new temp(b);
        }
}

//Setup Data
private void data(int address) {
        short tag = (short) ( (address & 0xF00) >>> 8 );
        short slot = (short) ( (address & 0x0F0) >>> 4 );
        short start = (short) (address & 0xFF0);
        short last = (short) (start + SIZE); short value = 0;
        for( short i = start; i < last; i++ ) {
                this.node[slot].setData( value++, this.memory[i] );
        }
        this.node[slot].setTag(tag);
        this.node[slot].setValid(true);
}
```