```
#include <iostream>
#include <string>
#include <fstream>
#include <vector>
#include <cstdio>          // for read
#include <cctype>          // for isspace
#include <stdexcept>       // for runtime_exception
using namespace std;

#include <unistd.h>         // for lseek

#include "ctimer.h"

class Reader
{
  public:
    virtual void read(istream& input) = 0;
    virtual int size()          const = 0;
    virtual const char * get(int k)  const = 0;
    virtual string sget(int k) const = 0;
};

class SimpleReader : public Reader
{
  public:
    SimpleReader() { }
    virtual void read(istream& input){
        string word;
        while (input >> word){
            myWords.push_back(word);
        }
    }

    virtual int size() const {
        return myWords.size();
    }

    virtual const char * get(int k) const {
        return myWords[k].c_str();
    }

    virtual string sget(int k) const {
        return myWords[k];
    }
  private:
    vector<string> myWords;
};

class RawReader : public Reader
{
  public:
    RawReader()
        : myBuffer(0),
          myWords(0),
          myCount(0)
    {

    }

    virtual void read(istream& input) {

        ifstream * ifp = dynamic_cast<ifstream*>(&input);

        if (ifp != 0){
            readFile(*ifp);
        }
        else {
            readStdin();
        }
```



```
    }

    virtual int size() const {
        return myCount;
    }

    virtual char * get(int k) const {
        return myWords[k];
    }

    virtual string sget(int k) const {
        return get(k);
    }

  private:

    void readFile(ifstream& input){
        filebuf * pbuf = input.rdbuf();

        // find size and set seek position to start of buffer

        long size = pbuf->pubseekoff(0, ios::end, ios::in);
        pbuf->pubseekpos(0, ios::in);

        // allocate buffer and fill it

        myBuffer = new char[size+1];
        pbuf->sgetn(myBuffer,size);

        input.close();
        findWords(size);
    }

    void readStdin(){

        // fd zero is for stdin, find size and set seek to start
        int fd = 0;
        long  size = lseek(fd,0,SEEK_END);
        lseek(fd,0,SEEK_SET);

        // allocate buffer and fill it

        myBuffer = new char[size+1];
        if (::read(fd,myBuffer,size) != size) {
            throw new runtime_error(string("error reading cin"));
        }
        findWords(size);
    }

    void findWords(int size){

        /**
         * state machine approach to wordness.
         * Look for space/non-space transition to find words
         */

        bool inWord = false;             // initially not in word
        int  count = 0;
        char * start = myBuffer;

        for(int k=0; k < size; k++) {
            if (isspace(myBuffer[k])) {
                if (inWord) {            // just finished a word
                    inWord = false;
                    myWords.push_back(start);
                    myCount++;
                    myBuffer[k] = '\0';  // mark end of word
                }
```

```
            }
            else {                          // not a space
                if (! inWord) {             // just started a word
                    inWord = true;
                    start = &myBuffer[k]; // remember start
                }
            }
        }
        if (inWord) {
            myWords.push_back(start);
            myCount++;
            myBuffer[size] = '\0';
        }
    }

    char *  myBuffer;
    vector<char *> myWords;
    int     myCount;
};

double stressRead(istream& input, Reader * r)
{
    CTimer timer;
    timer.Start();
    r->read(input);
    timer.Stop();
    return timer.ElapsedTime();
}

double stressAccess(Reader * r)
{
    CTimer timer;
    int size = r->size();
    timer.Start();

    for(int k=0; k < size; k++){
        const char * s = r->get(k);
    }
    timer.Stop();
    return timer.ElapsedTime();
}

void test(istream& input, Reader * r)
{
    double rtime = stressRead(input,r);
    cout << "time to read\t" << r->size() << " ";
    cout << rtime << " secs." << endl;

    double atime = stressAccess(r);
    cout << "time to access\t" << r->size() << " ";
    cout << atime << " secs." << endl;

    cout << endl;
}

int main(int argc, char * argv[])
{
    bool fromFile = false;
    ifstream input;
    if (argc > 1){
        fromFile = true;
        input.open(argv[1]);
    }

    Reader * reader = new SimpleReader();
    Reader * rreader = new RawReader();

    if (fromFile){
        test(input,reader);
```



```
        test(input,rreader);
    }
    else {
        test(cin,reader);
        test(cin,rreader);
    }
}
```