bool operator< (const Date& a, const Date& b) { return (a.getYear() < b.getYear() || (a.getYear() == b.getYear() && a.getMonth() < b.getMonth()) || (a.getYear() == b.getYear() && a.getMonth() == b.getMonth() && a.getDay() < b.getDay())); }

std::ostream& operator<< (std::ostream& ostr, const Name& n) { ostr << n.first() << “ “ << n.last();

return ostr; }

-----------------------------------------

p = &x; //store x’s memory position

\*p = 7; //change val @ p mem

p = q; //same pointer now

int \*p = q //Above, at init

**Pointer Iteration**

const int n = 10; double a[n];

double \*p;

for (p=a; p<a+n; ++p)

\*p = sqrt( p-a );

int \*a = new int[n];

for(int \*p = a; p < a+n; p++){

std::cin >> \*p;} //Read to ptr

//Coutptr

for(int i =0; i < n; i++)

cout << a[i] << endl;

**Dynamic 2D Array**

int\*\* a = new int\*[rows];

for (int i = 0; i < rows; i++) {

a[i] = new int[cols];

for (int j = 0; j < cols; j++) {

a[i][j] = int(i+1) / int (j+1);}}

**Create dynamic array**

int readInt;

int\* intArray = new int[MAX\_ARRAY\_SIZE];

std::ifstream input(filename);

while (input >> readInt) {

\*(intArray + \*numElements) = readInt;

\*numElements += 1;

}

**class** Foo {

public:

double x; int\* y;};

//Main

Foo a; a.x = 3.14159;

Foo \*b = new Foo;

(\*b).y = new int[2];

Foo \*c = b;

a.y = b->y;

c->y[1] = 7;

b = NULL;

#ifndef \_\_Class\_h\_

#define \_\_Class\_h\_

#endif

//Read until we get something that doesn’t match type

int score;

for (i=0; i<num\_homeworks && (in\_str >> score); ++i) { hw\_scores\_.push\_back(score); }

OR:

std::ifstream in\_str(fName);

std::string currLine;

while(in\_str >> currLine){

fileTxt.push\_back(currLine); }

class Clothing{

public:

Clothing(const std::string& cid);

const std::string& getID() const { return id; }

bool isDirty() const { return dirty; }

bool isClean() const { return !dirty; }

void Wash();

private:

std::string id;

bool dirty; };MOVING TO.CPP

#include "Clothing.h"

Clothing::Clothing(const std::string& cid){

id = cid; dirty = true;}

const std::string& Clothing::getID() const{

return id;}

bool Clothing::isDirty() const{

return dirty;}

void Clothing::Wash(){

if(!dirty){

std::cerr << id << " already clean" << std::endl;

return; } dirty = false;}

int i,\*\*j,k,l,\*m;

i = 0; j = new int\*[3];

j[0] = new int; j[1] = &i;

m = \*(j+1);

j[1] = &k;

k=10;

\*(j[0]) = 5; j[2] = j[0];

\*(j[0]) = 18; \*m = 4; l = 3;//GOES TO BELOW

Diagram

Description automatically generated

Diagram

Description automatically generated

int p = 20; int \*x = &p;

std::string \*y = new std::string[2];

y[0] = "ox"; y[1] = "cat";

y = new std::string;

\*y = "five"; int q[3]; q[0] = 1; q[1] = 3; q[2] = 2;

int \*z = &q[1];GOES TO ABOVE

#include <iostream> (read and write from console), <cmath> (math lbry, sqrt(), and abs() ), <cstdlib> (exit function)

std::cout, std::endl // console

if (condition) { statement };

else { statement };

double a[15]; //array with space for 15 doubles

a[5] = 3.14159 //array index start at 0

-----------------------------------------

hiStr.size();//size of string

std::str.substr (3,5); starts at index 3, stretches 5 chars

std::str.find(findME);

(str.find(thing) != std::string::npos) //If thing in string do this.

std::string c = “hi”+“ “ + “bro”;

std:string stuff(2, ‘.’); 🡪 “..”

char h[] = {‘H’, ‘e’, ‘l’, ‘l’, ‘o’};

// or char h[] = “Hello!”;

std::vector<double> d(2,2.33);

// {2.33, 2.33} ALTERNATIVELY

#include <algorithm>

std::sort(a.begin(), a.end()) //3rd sort function optional

**Files:**

std::ifstream inFile(“name”);

std::ofstream outFile(“name”);

AS PARAMETER BELOW🡪

std::ifstream &grade\_str)

if (!inFile) {}, if (!out\_str.good()) {}

write to file

inFile >> s1 >> s2 >> s3;

//reads items, no spaces?