using namespace std; **be careful of double vs int**  #include <iostream> for cin and cout

#include <cmath>

#include <string> for anything with a string, size, length, at append  **s.size()**

str.assign(string,inclusive start index, how many chars) str.insert(5, " are "); // Inserts " are " at 5th index of str

str.substr(index, length) returns substring starting at index w length char

str.erase (position of start, length); str.find(item, indx) starts at index indx.

str.replace(from indx, indxnum (non incl), subStr)

#include <cctype>

isdigit(), isupper(), islower(), isspace(), isalpha(), toupper(),tolower()

**if** (isalpha(ch)) //RIGHT && **if** (isalpha(ch) == **true**) // WRONG!!!!

cout.setf(ios::fixed); // double with fixed dp //to check for equality with two floats

cout.precision(2); **double** x, y;

cin.ignore(100, '\n'); After int before getline fabs(x - y) < 0.0001; //fabs is to take absolute value

**switch**(expression) { cout <<; cin>>;

**case** x: //x can only be an int

// code block

**break**;

**default**:

// code block}

The / operator performs floating-point division if at least one operand is a floating-point type.

Do{ \* / % higher precedence (left to right if same precedence)

…} while(expression) + - lower precedence (left to right if same precedence)

Evaluate ! before &&, && before ||, but can override precedence with ()

#include <cstring>;

**char** encodedMsg[] = "BAAANPALNEA";

This has 11 chars but array is 12 bc ‘\0’ (zero byte)

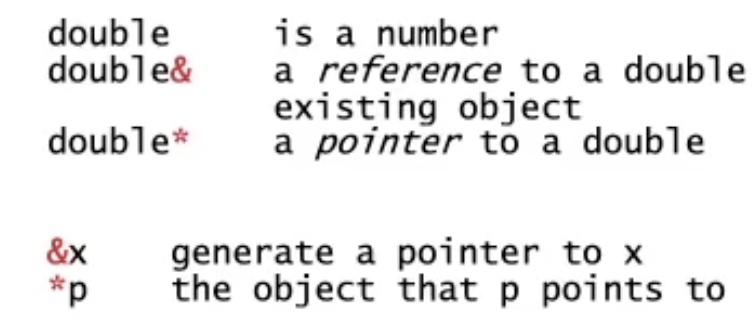
for(int i = 0; i <strlen(encodedMsg); i++)

strlen(encodedMsg) = 11

strcpy(destStr, sourceStr)

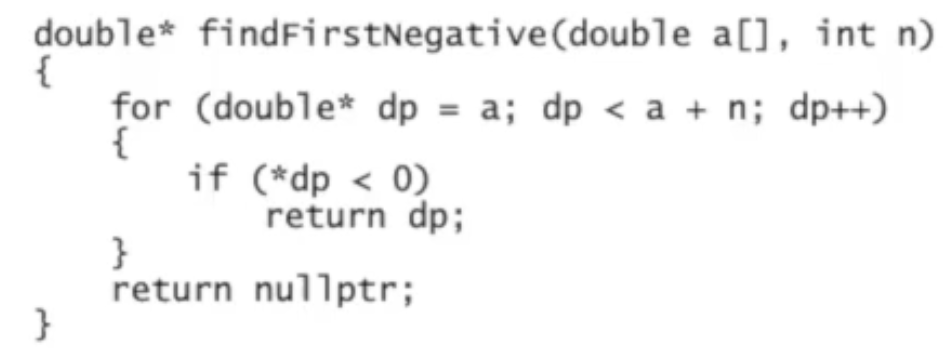
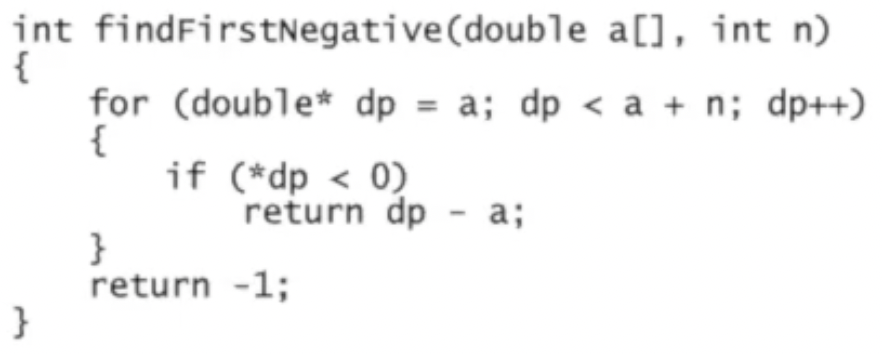


**0 IS FALSE**



double a[MAXSIZE] = …

for(double p\* = a; p < a+MAXSIZE; p++)



Arrays:

Passing in arrays in functions

int sum(const int a[], int n) {...} // allowed to pass in regular int[] and const int[] data

void setAll(int a[], int n) {...} // only int[] can be passed; const int[] is compile error

You must define size for 2D+ arrays from 2nd dimension beyond in the parameters:

int calc(char a[][SIZE], int n) {...}

CStrings:

#include <cstring>

‘\0’ zero byte to terminate C-string

char t[10] = {‘h’, ‘e’, ‘l’, ‘l’, ‘o’, ‘\0’};

char t[10] = "hello"; // ‘\0’ is implied, tacked on at t[5]

You cannot assign or concatenate C-strings with regular = or +=. Use these methods:

strlen(t) // length of C-string, excluding ‘\0’

strcpy(s, t) // copy string t to string s (strcpy(dest, src)), adjust ‘\0’ accordingly

strcat(s, \!!!") // concatenate \!!!" to the C-string s and add ‘\0’ to the end

strcmp(s, t) // if s < t return negative, s == t return 0, s > t return positive

DYNAMICALLY CREATE OBJECT:

**if** (rabbitCount() >= MAXRABBITS)

**return** **false**;

m\_rabbits[m\_nRabbits] = **new** Rabbit(**this**, r, c);

m\_nRabbits++;

**return** **true**;

DESTRUCTOR