Friday, December 7, 2018 3:13 PM

Table of Contents - Sections Pages

    - isDigit
    - isPalindrome
    - Reverse
    - Strcat***
    - Strcpy
    - GetInt***
    - Panagram***
    - Longest Sequence***
    - Strlen***
    - Add***
    - Last Occur***
    - Strcmp***
    - Haystack***
    - ShiftY***
    - Passage.txt and getWord***
    - indexOf***
    - No Comments***
- Dev
    - Planning
    - Chad Flyer

Welcome

Sunday, December 2, 2018 2:20 PM

key 10

5 10 15 20 25 30 35 40
0 1 2 3 4 5 6 7

THE PATHS OF PROGRAMMING:

https://coggle.it/diagram/W1gOaQ8IlqhFzrmg/t/programming/c4e9a26d190c5cea8cb99b1ac3b082da0f7fa8f03f43d879aef7aabc0ba29c6f

THE PLAN

Teach the basic IDEAS, CONCEPTS QUICKLY

Apply to questions

THE PROBLEM : A single question can involve multiple topics

MATERIALS

Because of time

Goal – LESS Note taking, Focus on Problem Solving

- ONE NOTE EXPORTED PDFS → Detailed step by step Explanations with illustrations & links to further Resources → MyCodeSchool, HackerRank, Geeks for Geeks
- C++ CODE FILES WHERE APPLICABLE.
- SHARED VIA GOOGLE DRIVE.

Still Not for self study Purely Supplementary

Screenshot & paste in ONE NOTE.

COGGLE

https://drive.google.com/open?id=19LYEW6AFDeekPsDfCO3VqlUTj9JbU6Nk

This is not my preferred method of teaching.
But I have to work with time.
And it will provide you with a hopefully effective STRUCTURE of revising the concepts.
It may seem boring @ first,
BEAR WITH ME :)

How to approach a question ?

- Read
- Understand
- Answer in English (Pseudocode)
- Translate English to code
- Code on a computer

Learn the Process
Not the code

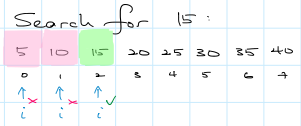
DIAGRAMS!!!!!!
VISUALIZE!!!!!

Linear Search

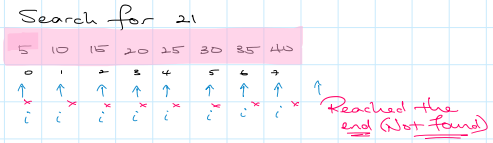
Sunday, December 2, 2018 3:03 PM

Start at the begining of the array
Continue until found or end of array

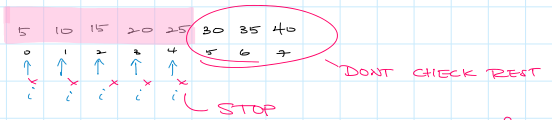
EXAMPLE : Search for 15 :

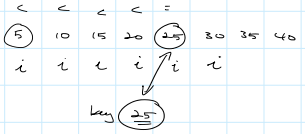


Search for 21



Special Case when dealing with SORTED arrays
Searching for 21





ALGORITHM

```
int linearSearchUnsorted(int A[], int n, int key){
    int i = 0;                      // start at the beginning of the array
    while(i < n && A[i] != key)     // while (we haven't reached the end of the array) AND our current value is not key:
        i++;                            //go to the next location
    if(A[i] == key)                 // if current value that we are examining is key:
        return i;                       // we are done, return the LOCATION of key
    else                            // else we went outside the bounds of the array (i is not less than n):
        return -1;                      //not found
}


int linearSearchSorted(int A[], int n, int key){
    int i = 0;                      // start at the beginning of the array
    while(i < n && A[i] < key)      // while (we haven't reached the end of the array) AND our current value is not key:
        i++;                            //go to the next location
    if(A[i] == key)                 // if current value that we are examining is key:
        return i;                       // we are done, return the LOCATION of key
    else                            // else we went outside the bounds of the array (i is not less than n):
        return -1;                      //not found
}
```

Binary Search

Tuesday, December 4, 2018 9:00 PM

HUGE Array must be sorted

REPEAT:
- Find middle
- Determine what portion key lies in
- Shrink the portion to be searched

UNTIL FOUND or DOES NOT EXIST

EXAMPLE 1

SEARCH FOR 10

5	10	15	20	25	30	85	40
0	1	2	3	4	5	6	7

FIND MIDDLE — $(f + l)/2 = 3$ $(0+7)/2 = 3$

5	10	15	20	25	30	85	40
0	1	2	3	4	5	6	7
f			m				l

f [0] l [7] m [3]

10 is to the left of 20, so shrink the array to the LEFT

5	10	15	20	25	30	85	40
0	1	2	3	4	5	6	7
f		l					

COULD BE HERE — DEFINITELY NOT HERE!

FIND MIDDLE — $(0+2)/2 = 1$

5	10	15	20	25	30	85	40
0	1	2	3	4	5	6	7
f	m	l					

f [0] l [2] m [1]

FOUND! SO STOP

EXAMPLE 2

FIND 40

FIND MIDDLE (SAME) — $(0+7)/2 = 3$

5	10	15	20	25	30	85	40
0	1	2	3	4	5	6	7
f			m				l

f [0] l [7] m [3]

40 is to the right of 20 so shrink to the right

5	10	15	20	25	30	85	40
0	1	2	3	4	5	6	7
				f			l

FIND MIDDLE — $(4+7)/2$

5	10	15	20	25	30	85	40
0	1	2	3	4	5	6	7
				f	m		l

f [4] l [7] m [5]

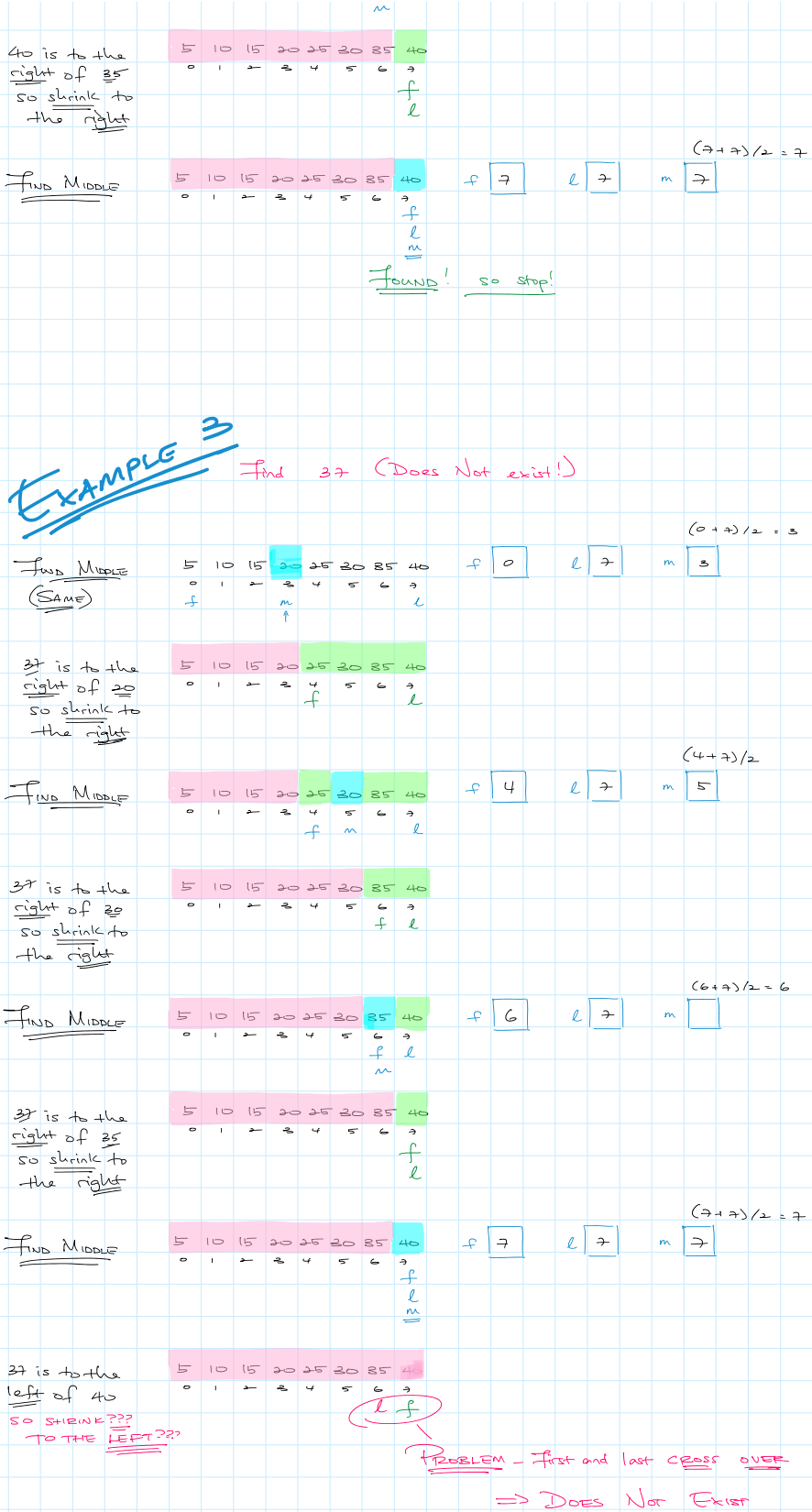
40 is to the right of 30 so shrink to the right

5	10	15	20	25	30	85	40
0	1	2	3	4	5	6	7
						f	l

FIND MIDDLE — $(6+7)/2 = 6$

5	10	15	20	25	30	85	40
0	1	2	3	4	5	6	7
						f	l

f [6] l [7] m [ ]



ALGORITHM

```
int binarySearch(int A[], int n){
    int f = 0;                    // start first at the beginning of the array
```

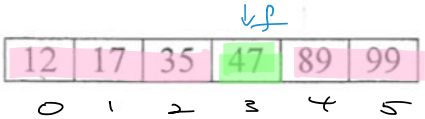
```
int binarySearch(int A[], int n){
    int f = 0;                  // start first at the beginning of the array
    int l = n-1;                // start last at the end of the array
    while(f <= l){              // while first and last have not crossed over
        int m = (f+l)/2;            // calculate middle
        if(key < A[m])              // if key is to the left of the middle value
            l = m-1;                    // shrink the array to the left
        else if(key == A[m])        // else if key is equal to the middle value
            return m;                   // we are done, key found
        else                        // else the only other possibility is key is to the right of middle value
            f = m+1;                    // so shrink array to the right
    }
    return -1;                  // if first and last cross over then key is not found
}
```

Question***

Wednesday, December 5, 2018 3:01 AM

17-18 S2 COMP1602 Final

(b) The array below is sorted in *ascending* order:



Using a binary search technique, how many comparisons are needed to find the following keys?

i. 47
ii. 55

Show all working. [5 marks]

Total marks: 10

	f	l	m	Comparison
1	0	5	2	47 > 35 => f = m+1
2	3	5	4	47 < 89 => l = m-1
3	3	3	3	47 = 47 => found

Question***

Wednesday, December 5, 2018 3:18 AM

16-17 S2 COMP1602 Final

b. The array below is sorted in *descending* order.

34	32	29	23	17	12	8	6	3	1
0	1	2	3	4	5	6	7	8	9

Using a binary search technique, how many comparisons are need to find the following keys?

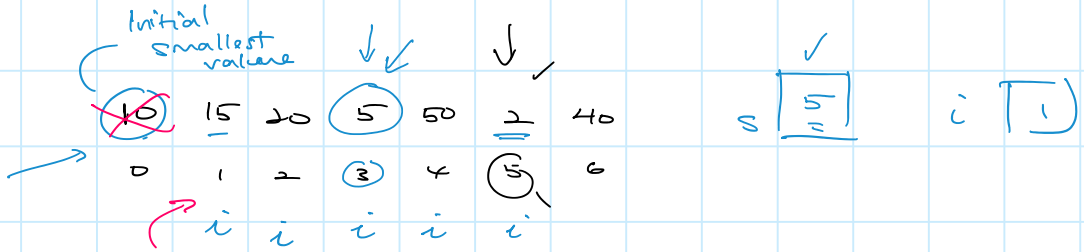
i. 29
ii. 50

Show all working. [4]

f l m Comparison

Finding location of smallest element in an array

Wednesday, December 5, 2018 11:51 AM



Assume s is location 0

```
int s = 0;                          - Assumption
for (int i = 1; i < n; i++)
    if (A[i] < A[s])
        s = i;
```

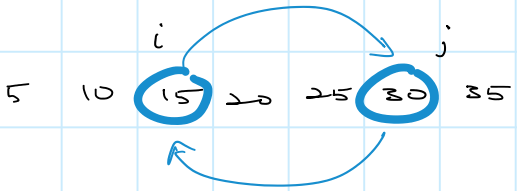
Check the rest of the array!

s is NOT the VALUE in the array

it is the LOCATION OF the value.

Swapping values in an array

Wednesday, December 5, 2018 12:02 PM




```
int temp = A[i];
A[i] = A[j];
A[j] = temp;
```

Selection Sort

Sunday, December 2, 2018 3:03 PM

IDEAS

The Array can be thought about as a SORTED and UNSORTED portion

ON EACH "PASS" the Algorithm "SELECTS" the location of the smallest or largest element from the unsorted portion & puts it where it belongs. (At the end of the sorted portion)

Sorted Portion Grows

Unsorted Portion Shrinks

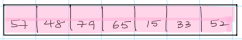
Build the algorithm through understanding, DO NOT MEMORIZE!

EXAMPLE

Kali Pg 171.

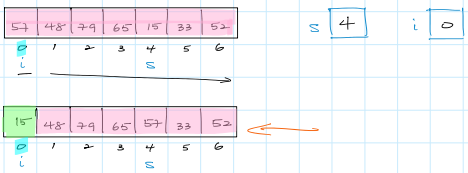
PASS (0)

Entire Array is considered unsorted



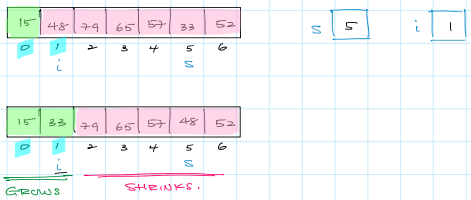
PASS (1)

- ✓ Select the location of the smallest element in the unsorted portion.
- ✓ Put the value in S in its place swap S and i



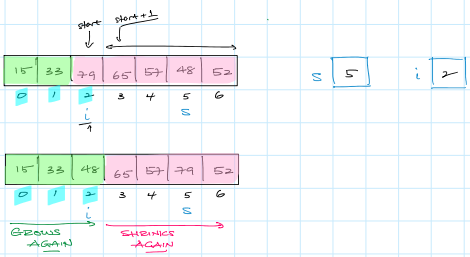
PASS (2)

- ✓ Select the location of the smallest element in the unsorted portion.
- ✓ Put the value in S in its place swap S and i



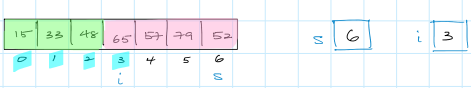
PASS (3)

- ✓ Select the location of the smallest element in the unsorted portion.
- ✓ Put the value in S in its place swap S and i



PASS (4)

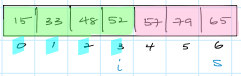
- ✓ Select the location of the smallest element



in the unsorted portion.

✓ Put the value in s in its place

swap s and i.



PASS (5)

✓ Select the location of the smallest element in the unsorted portion.

✓ Put the value in s in its place

swap s and i.

s 4 i 4

i, s swap

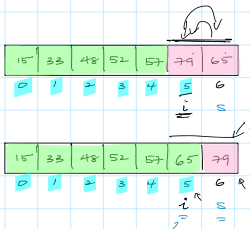
No CHANGE

PASS (6)

✓ Select the location of the smallest element in the unsorted portion.

✓ Put the value in s in its place

swap s and i.



s 6 i 5

DONE!

i takes values from 0 → 5

first

last location − 1

n − 2

ALGORITHM

```
void swap(int A[], int i, int j){
    int temp = A[i];
    A[i] = A[j];
    A[j] = temp;
}


int findSmallestLocationInUnsortedPortion(int A[], int start, int n){
    int s = start;                          // assume that the start location has the smallest value in unsorted portion
    for(int i = start+1; i < n; i++){       // check the REST of the unsorted portion for any value that may be smaller
        if(A[i] < A[s]){                        // if i find some values less than what i currently believe to be the smallest
            s = i;                                  // then update smallest location
        }
    }
    return s;
}


void selectionSort(int A[], int n){
    // start i at zero
    // we write i < n-1 because the last step automatically sorts the last 2 locations
    for(int i = 0; i < n-1; i++){
        int s = findSmallestLocationInUnsortedPortion(A, i, n);
        swap(A, i, s);
    }
}
```

Insert in place

Wednesday, December 5, 2018 12:00 AM

We have an array that is already sorted

We want to insert a new element into the array so we maintain the sorted order.

EXAMPLE Insert 10 into the array below.

5	15	20	25	30	35	40	
0	1	2	3	4	5	6	7

Start at the end of the array

Shift value to make space until

either (1) we reach the start of the array

or (2) We find where key should be inserted

Duplicates will exist temporarily.

5	15	20	25	30	35	40	40
0	1	2	3	4	5	6	7
						j	j+1

j = 6 key = 10

5	15	20	25	30	35	35	40
0	1	2	3	4	5	6	7
					j	j+1	

j = 5

5	15	20	25	30	30	35	40
0	1	2	3	4	5	6	7
				j	j+1		

j = 4

5	15	20	25	25	30	35	40
0	1	2	3	4	5	6	7
			j	j+1			

j = 3

5	15	20	20	25	30	35	40
0	1	2	3	4	5	6	7
		j	j+1				

j = 2

5	15	15	20	25	30	35	40
0	1	2	3	4	5	6	7
	j	j+1					

j = 1

5	15	15	20	25	30	35	40
0	1	2	3	4	5	6	7
j	j+1						

j = 0

PUT 10 HERE!

5	10	15	20	25	30	35	40

✓ DONE

ALGORITHM

```
void insertInPlace(int A[], int n, int key){
    int j = n-1;
    while(j >= 0 && key < A[j]){
        A[j+1] = A[j];
        j--;
    }
    A[j+1] = key;
}
```

Insertion Sort

Monday, December 3, 2018 5:01 PM

WARNING

Slightly more Complicated topic ahead
You might get confused the first time around.
Don't worry.
Remain Calm.
Trust in the process...

IDEAS

The Array can be thought about as a SORTED and UNSORTED portion

KEY
- Assume the first element is already "SORTED"
- INSERT the next element from the unsorted portion, into its correct place in the sorted portion.

Sorted Portion Grows
Unsorted Portion Shrinks

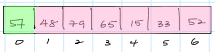
Build the algorithm through understanding, DO NOT MEMORIZE!

EXAMPLE

Kali pg 174-175

PASS (0)

The first element is considered "pre sorted" by default



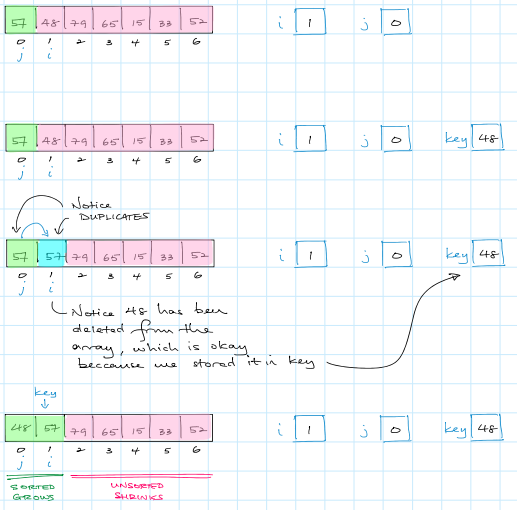
PASS (1)

Insert the next element from the unsorted portion into its right place in the sorted portion.

Save the value in i in key Because it will be temporarily deleted from the array.

Shift the value from the sorted portion up to make space for the new value to be inserted

Repeat this process until we (j+1) have found the appropriate spot to put the new value. (key)

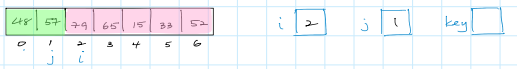


THIS IS THE END OF PASS 1

PASS (2)

Location (i)

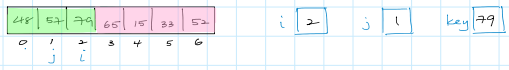
Insert the next element from the unsorted portion into its right place in the sorted portion.



Save i in key Because it will be temporarily deleted from the array

WE WOULD NORMALLY Shift the value from the sorted portion up to make space for the new value to be inserted BUT 79 is already where it needs to be

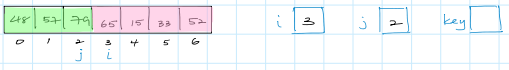
put the new value (key) where it should be (j+1)



THIS IS THE END OF PASS 2

PASS (3)

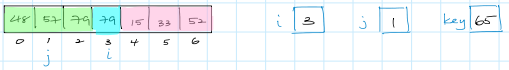
Insert the next element from the unsorted portion into its right place in the sorted portion.



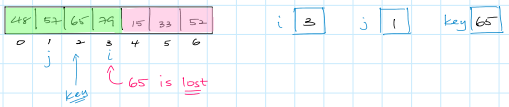
Save i in key Because it will be temporarily deleted from the array

Shift the value from the sorted portion up to make space for the new value to be inserted

DECREMENT j



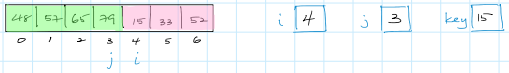
Repeat this process until we have found the appropriate (j+1) spot to put the new value. (key)



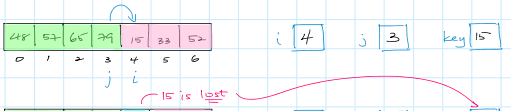
PASS (4)

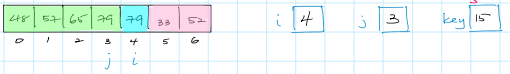
Insert the next element from the unsorted portion into its right place in the sorted portion.

Save i in key Because it will be temporarily deleted from the array

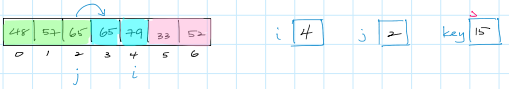


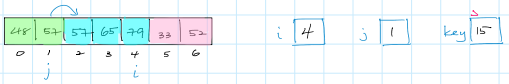
Shift the value from the sorted portion up to make space for the new value to be inserted

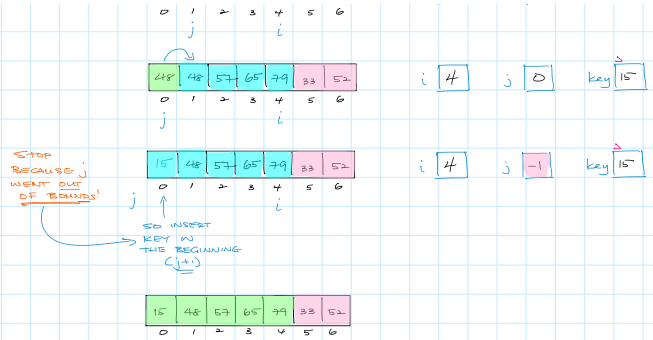




Repeat this process until we have found the appropriate (j+1) spot to put the new value. (key)

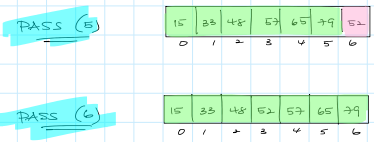






THIS IS THE END OF PASS (4)

I hope you guys get the idea by now because i am getting really tired with all this writing. So I'll just write out the states for the rest.



In exam, this is sufficient when asked to write the passes of the algorithm. You still NEED to understand the detailed process I outlined above. because it helps when developing the Algorithm

DO NOT MEMORIZE THE CODE
UNDERSTAND THE PROCESS

ALGORITHM

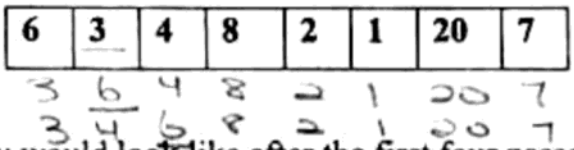
```
void insertInPlace(int A[], int n, int key){
    int j = n-1;
    while(j >= 0 && key < A[j]){
        A[j+1] = A[j];
        j--;
    }
    A[j+1] = key;
}


void insertionSort(int A[], int n){
    for(int i = 1; i < n; i++){
        cout << "PASS #" << i << ":\t";
        insertInPlace(A, i, A[i]);
        printArray(A, n);
    }
}
```

Question***

Wednesday, December 5, 2018 3:54 AM

16-17 S3 COMP1602 Final

(b) Assume you have an integer array as shown below and want to sort it in **ascending** order.



(i) Show what the array would look like after the first four passes of an *insertion sort*. [3]

(ii) Write a function **`insertionSort`** that accepts an array of integers **`arr`** and its size, **`size`** and sorts the array in ***ascending*** order. [7]

Sort Array of Date Structs***

Wednesday, December 5, 2018 3:27 AM

COMP1404_2_16

(b) You are given a *compare* function with the following prototype:

```
int compare(Date d1, Date d2);
```

compare returns -1 if *d1* comes before *d2*, 0 if *d1* is the same as *d2*, and 1 if *d1* comes after *d2*.

You are also given an array, *dates*, of Date structs, containing *numDates* elements. Write code to sort the array in ascending order of date using the **insertion sort** algorithm. *[5 marks]*

Sort C-Strings***

Wednesday, December 5, 2018 3:41 AM

b) You are given the following declarations:

```
typedef struct {
     char id[5];
     int value;
} Item;

Item series[25];
```

Assume that data have been stored in the first ***n*** elements of the array ***series***. Write code to sort the data in the array in descending order by ***id***, using the insertion sort algorithm. [5]

Zoo Keeper Question***

Wednesday, December 5, 2018 3:46 AM

16-17 S3 COMP1602 Final

A zoo keeper wishes to maintain some information about the animals at his small zoo. He only has one of each type of animal. For each animal, the animal name is stored (e.g. MONKEY), the number of immunization shots that animal obtained so far (integer) and the cost price (double) of the animal (e.g. $5000.50). A text file, **animals.dat**, contains a set of records for each animal at the zoo. There are no more than 50 entries in the file. Also, all text data in the file are in uppercase. Data is terminated by a line containing $$$$$ only. Assume that all animal names are unique, i.e. there are no duplicate animal names in the file. All animal names are single words.

Sample Data
MONKEY 5 5000.50
SQUIRREL 2 100.00
IGUANA 0 200
$$$$$

(a) Write code to read the data from the text file into an array of structs, **animalsArr**, using the following restriction: whenever a record is read, it must be inserted into the array so that the array is always in ascending order of animal name.

Show all relevant declarations. [12]

(b) After the array has been loaded, write code that prompts the user for an animal name and searches the array for that name using a **binary search** technique. If the name exists in the array, return the number of immunization shots received so far, otherwise, return -1.

Note: Assume that the user's input is in uppercase. [8]

Total Marks for Question 1 is 20

Bubble Sort??

Wednesday, December 5, 2018 2:29 AM

Merge***

Wednesday, December 5, 2018 3:37 AM

comp1404_2_14
16-17 S3 COMP1602 Final

2.

(a) An array **A** of size **n** is filled with integers in descending order. An array **B** of size **m** is filled with integers in ascending order. An array **C** is of type integer and sized **m+n**. Write a function **merge** that accepts **A**, **B**, **C** and their sizes, and merges the contents from **A** and **B** into **C** in ascending order. **Your code must take advantage of the way the numbers are arranged in A and B.** [6]

3. An integer array **A** contains **aSize** positive integer values. This array is sorted in **ascending** order. An integer array **B** contains **bSize** positive integer values. This array is also sorted in **ascending** order. Write code to merge the integers in **A** with the integers in **B**, and place the result of the merge into another integer array, **C**, such that the integers in **C** are in **descending** order.

Show all relevant declarations.

Note: Use arrays sizes of 50 for each of **A** and **B** and an array size of 200 for **C** when making the declarations. [10]

Question***

Wednesday, December 5, 2018 3:02 AM

17-18 S2 COMP1602 Final

3) (a) The array, *urban*, needs to be sorted in *ascending* order.

urban	Anon	Mila	Kamryn	Julia	Julia	Amara

Starting with the contents of *urban* as shown above, *draw* the modified array:

i. After each of the first three (3) passes of insertion sort;
ii. After each of the first three (3) passes of selection sort.

Show all working. [5 marks]

Question***

Wednesday, December 5, 2018 3:18 AM

16-17 S2 COMP1602 Final

4. a. The array, num, needs to be sorted in *ascending* order.

num	32	8	29	6	17	32	3	12	8	23

Starting with the contents of num as shown above, *draw* the modified array:

i. After each of the first three (3) passes of insertion sort;
ii. After each of the first three (3) passes of bubble sort.
iii. If the array was already sorted, would insertion sort or bubble sort be more efficient?

Show all working. [6]

Selection Sort

(0) 32 8 29 6 17 32 3 12 8 23

(1) 3 8 29 6 17 32 32 12 8 23

(2) 3 6 29 8 17 32 32 12 8 23

(3) 3 6 8 29 17 32 32 12 8 23

(4) 3 6 8 8 17 32 32 12 29 23

Review of mod operator

Sunday, December 2, 2018 3:04 PM

✔ % gives the remainder from division

RULE x % n ∈ ?

TABLE OF MODS.

Row % Col	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	0	0	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
3	0	1	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
4	0	0	1	0	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
5	0	1	2	1	0	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6	0	0	0	2	1	0	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6
7	0	1	1	3	2	1	0	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7
8	0	0	2	0	3	2	1	0	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
9	0	1	0	1	4	3	2	1	0	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9
10	0	0	1	2	0	4	3	2	1	0	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
11	0	1	2	3	1	5	4	3	2	1	0	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11
12	0	0	0	0	2	0	5	4	3	2	1	0	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12
13	0	1	1	1	3	1	6	5	4	3	2	1	0	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13
14	0	0	2	2	4	2	0	6	5	4	3	2	1	0	14	14	14	14	14	14	14	14	14	14	14	14	14	14	14	14
15	0	1	0	3	0	3	1	7	6	5	4	3	2	1	0	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15
16	0	0	1	0	1	4	2	0	7	6	5	4	3	2	1	0	16	16	16	16	16	16	16	16	16	16	16	16	16	16
17	0	1	2	1	2	5	3	1	8	7	6	5	4	3	2	1	0	17	17	17	17	17	17	17	17	17	17	17	17	17
18	0	0	0	2	3	0	4	2	0	8	7	6	5	4	3	2	1	0	18	18	18	18	18	18	18	18	18	18	18	18
19	0	1	1	3	4	1	5	3	1	9	8	7	6	5	4	3	2	1	0	19	19	19	19	19	19	19	19	19	19	19
20	0	0	2	0	0	2	6	4	2	0	9	8	7	6	5	4	3	2	1	0	20	20	20	20	20	20	20	20	20	20
21	0	1	0	1	1	3	0	5	3	1	10	9	8	7	6	5	4	3	2	1	0	21	21	21	21	21	21	21	21	21
22	0	0	1	2	2	4	1	6	4	2	0	10	9	8	7	6	5	4	3	2	1	0	22	22	22	22	22	22	22	22
23	0	1	2	3	3	5	2	7	5	3	1	11	10	9	8	7	6	5	4	3	2	1	0	23	23	23	23	23	23	23
24	0	0	0	0	4	0	3	0	6	4	2	0	11	10	9	8	7	6	5	4	3	2	1	0	24	24	24	24	24	24
25	0	1	1	1	0	1	4	1	7	5	3	1	12	11	10	9	8	7	6	5	4	3	2	1	0	25	25	25	25	25
26	0	0	2	2	1	2	5	2	8	6	4	2	0	12	11	10	9	8	7	6	5	4	3	2	1	0	26	26	26	26
27	0	1	0	3	2	3	6	3	0	7	5	3	1	13	12	11	10	9	8	7	6	5	4	3	2	1	0	27	27	27
28	0	0	1	0	3	4	0	4	1	8	6	4	2	0	13	12	11	10	9	8	7	6	5	4	3	2	1	0	28	28
29	0	1	2	1	4	5	1	5	2	9	7	5	3	1	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	29
30	0	0	0	2	0	0	2	6	3	0	8	6	4	2	0	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Random number generation

Wednesday, December 5, 2018 2:04 AM

Always seed!

Rule: x % n gives output in the range 0 -> n-1

/ Generate a number between

0 and 9

0 and 99

0 and 13

1 and 10

1 and 100

1 and 14.

A year in the range [1997, 2018]

FORMULA TO GENERATE VALUES IN RANGE [LOW, HIGH]

I have an array of size n.
Generate a random number in the range [32, 65]
and store it in a random location of the array.

Lotto Ticket***

Wednesday, December 5, 2018 3:08 AM

16-17 S2 COMP1602 Final

3. Purchasing a Lotto ticket requires one to pick 7 numbers from the numbers 1 to 40. Write a program to randomly generate and print the numbers for 5 Lotto tickets, that is 5 sets of 7 numbers each, one set per line. No number is to be repeated in any of the tickets that is, exactly 35 of the 40 numbers must be used. If a number, say p, is generated which has been used already, the *first* unused number *after* p is used. Assume that 1 follows 40.
For example, if 15 is generated but has been used already, 16 is tried but if this has been used, 17 is tried and so on until an unused number is found. [10]

Total marks: 10

Gambler's bet***

Wednesday, December 5, 2018 3:30 AM

COMP1404_2_16
COMP1404_1_14
16-17 S3 COMP1602 Final

4. A gambler bets $10 to play the following game:

She throws two 6-sided dice. If the sum of the two numbers thrown is even, she loses her bet. If the sum is odd, she draws a card from a deck of 13 distinct cards from the suit of hearts. If she draws an Ace, 3, 5, 7, or 9, she is paid three times the value of the card plus her $10 bet. If she draws any other card, she loses her bet. Note that we can assign 1 to the Ace of hearts, 2 to the two of hearts, and so on up to 13 to the King of hearts.

Write code to simulate the playing of 100 games and print the average amount won by the player at the end of the simulation.

NB: The player's bet must be considered when calculating the winnings. *[10 marks]*

Total Marks for Question 4 is 10

b) A player throws two 6-sided dice. If the sum of the two numbers thrown is even, he loses the game. If the sum is odd, he draws a card from a standard pack of 52 playing cards. If he draws an ace, 3, 5, 7 or 9, he is wins the value of the card plus $5 (ace counts as 1). If he draws any other card, he loses. Write code to simulate the playing of 200 games and print the average amount won by the player at the end of the simulation.

[6]

4. In a game called "game X", a user pays $20 to play a game.
For each game, a die is tossed. If the number is odd, the user loses (i.e. the game ends and the player loses his bet). If the number is even, a card is drawn from a standard deck of 52 cards. If hearts is drawn, the user gets
$(2^n + 100)$ dollars where **n** was the value on the die toss. If the user does not draw hearts, he only receives $5.

[PTO]

Sample scenarios

DIE TOSS	CARD PULL	WINNINGS
1	None	0
2	Spades	$5
6	Hearts	$2^6+100 = \$164$

Assume that Jamie decides to play 20 games. Write code to simulate her 20 games of "game X". At the end, print her total profit or loss from playing the games. Note that each game costs $20.

Further notes:
A standard deck of cards contains 13 hearts, 13 spades, 13 clubs and 13 diamonds.
A die can land on any of the following when thrown: 1, 2, 3, 4, 5, 6.

[10]

Cricket***

Wednesday, December 5, 2018 2:09 AM

(b) In the game of cricket, a batsman based on his past statistics of play, has the following chance of scoring runs:

Runs	Chance
6	5%
5	3%
4	12%
3	5%
2	25%
1	30%
0	13%
out	7%

Write a function to simulate and print the runs made by this batsman until he is **out** or has attained **400 runs**. Your function must also print a summary of the number or 6's, 5's, 4's, 3's, 2's and 1's made by the batsman and his total number of runs. [8]

Biased Die***

Wednesday, December 5, 2018 2:09 AM

comp1404_2_14

(c) A certain 8-sided die is weighted such that 7's and 8's come up half as likely and 4's comes up twice as often as any other number. Write a function to simulate 500 throws of this die and output the frequency with which each number occurs. [6]

Total Marks 20

Defects***

Wednesday, December 5, 2018 2:09 AM

17-18 S1 COMP1602 Final Kali

In the manufacture of electric bulbs, the probability that a bulb is defective is 0.01. Write a program to simulate the manufacture of 5000 bulbs, indicating how many are defective. Ensure that each time your program is run, a different answer could be obtained. [3]

Marbles***

Wednesday, December 5, 2018 3:57 AM

17-18 S1 COMP1602 Final

A barrel contains a large number of marbles of various colours. Marbles are picked at random from the barrel. There is a 50% chance of picking a red marble, a 20% chance of picking a blue marble, a 20% chance of picking a green marble, a 5% chance of picking a yellow marble and a 5% chance of picking a black marble. Assume that these percentages do not change after a marble is picked.

Write a program to perform 25 simulations of the picking of marbles until at least one marble of each colour is picked. For each simulation, print the number of marbles picked. Also print the average number of marbles picked per simulation. [7]

Generating Cards

Wednesday, December 5, 2018 2:09 AM

Spelling a word

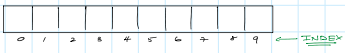
Wednesday, December 5, 2018 2:10 AM

Quick Recap

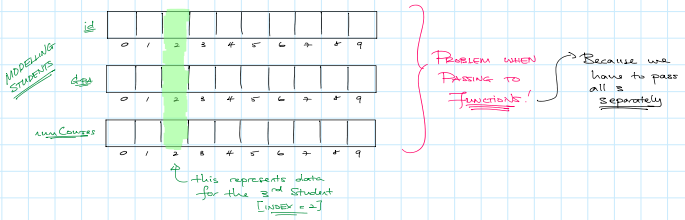
- ✓ A variable is a box — Always VISUALIZE the box

- ✓ We have different types of boxes
  - int
  - float / double
  - bool
  - string
  - char

- ✓ Arrays are many of these boxes squished together:

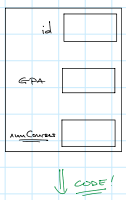


- ✓ Parallel Arrays have limitations.



Structs MODEL Real-World "Things"

A Student Struct [A Struct is like a NEW DATA TYPE].



⇓ CODE!

Declaring our struct (NEW DATA TYPE). — THE NAME OF OUR NEW DATA TYPE! (think about this as int or float because we have just made a New Data type).

```
struct Student {
    int id;
    float GPA;
    int numCourses;
};
```

Fields within the Struct describe attributes of the Real world thing.

Making a Variable

```
Student s;          // think about this like "int num"
Student s1;
Student student;
```

DATA TYPE: Student — VARIABLE NAME: s, s1, student

Fields (Attributes) are accessed by the dot operator

```
student.id = 100;
student.GPA = 4.0;
student.numCourses = 5;
```

THATS IT!!

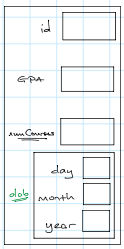
Everything else involving structs simply BUILDS UPON previous concepts / Abstractions.

Nested Structs

The Date Struct


```
struct Date{
    int day, month, year;
};
```

Nesting Date in Student

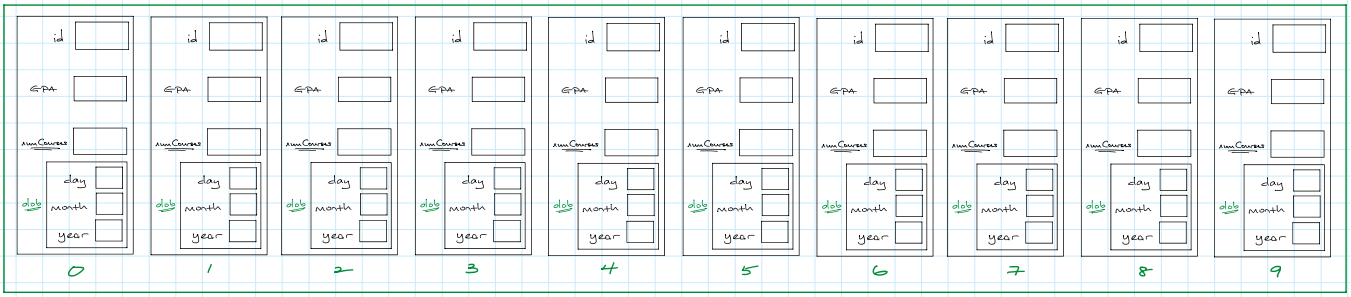



```
struct Student {
    int id;
    float gpa;
    int numCourses;
    Date dob;
};
```

An Array of Structs

→ Must be able to visualize!

```
Student students[10];
```



Visualizing Structs Train Schedular Example (Easy)

Sunday, December 2, 2018 4:35 PM

Visualizing Structs Train Scheduler Exercise (Hard)

Sunday, December 2, 2018 4:12 PM

COMP1404_1_14

A Train Scheduling System can be implemented using the declarations below:

```
#define MAXLENGTH 20
#define TRAINCAPACITY 200
#define MAXENTRIES 5000

typedef struct{
        int day, month, year;
}Date;

typedef struct{
        int hr, min, sec;
}Time;

typedef struct{
        int id;    //No duplicates allowed
        char name[MAXLENGTH];
}Passenger;

typedef struct{
        int id;     //No duplicates allowed
        char type;  //S = Slow Train ; F = Fast Train*/
        int passengerCount;
        Passenger listing[TRAINCAPACITY]; /*Sorted by Passenger id*/

}Train;

typedef struct{
        int id;    //No duplicates allowed
        char city[MAXLENGTH];
}Station;

typedef struct{
        Date d;
        Time arrival_t;
        Time departure_t;
        Train t;
        Station s;
}Schedule;

typedef struct{
        int scheduleCount;
        Schedule listing[MAXENTRIES]; /*Unordered Listing*/
}TrainScheduler;
```

Given these declarations, write efficient functions to implement the following operations:

i. `Station createStation(int id, char city[])`
/* returns a new Station with the given id and city*/ [3]

ii. `int hasSpace(Train t, int n)`
/* returns 1 if Train t has space for n Passengers;
otherwise return 0.*/ [4]

iii. `int containsPassenger(Train t, Passenger p)`
/* returns 1 if Passenger p is on Train t; otherwise return 0. */ [5]

iv. `int validateFastTrip(TrainScheduler ts, char src[], char dest[])`
/* returns 1 if there exists a fast Train that goes through the "src" city and the "dest" city;
otherwise returns 0.*/ [8]

Date***

Wednesday, December 5, 2018 3:20 AM

Add in ones from coggl too

COMP1404_2_16

2. Consider a Date struct which is declared as follows:

```
struct Date {
        int year;
        int month;
        int day;
};
```

(a) Write a function, *displayDate*, which accepts a Date struct as a parameter and displays the struct in the format: day/month/year. The following is the prototype of the *displayDate* function:

```
void displayDate(Date d);
```

[2 marks]

(b) You are given a *compare* function with the following prototype:

```
int compare(Date d1, Date d2);
```

compare returns -1 if *d1* comes before *d2*, 0 if *d1* is the same as *d2*, and 1 if *d1* comes after *d2*.

You are also given an array, *dates*, of Date structs, containing *numDates* elements. Write code to sort the array in ascending order of date using the **insertion sort** algorithm. *[5 marks]*

(c) Write a function *nextDay* with the following prototype:

```
Date nextDay(Date d);   // assume February always has 28 days
```

nextDay accepts a Date struct, *d*, as a parameter and returns the date of the day that comes immediately after *d*. For example, the date of the day that comes after "April 30, 2016" is "May 1, 2016" and the date of the day that comes after "December 31, 2015" is "January 1, 2016".

NB: Your *nextDay* function should assume February always has 28 days. *[7 marks]*

(d) Write a function, *interval* with the following prototype:

```
int interval(Date d1, Date d2);
```

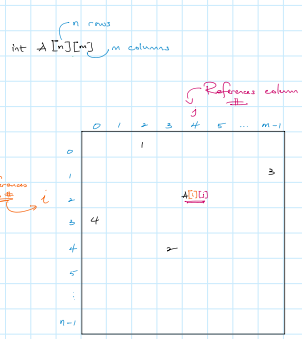
interval accepts two Date structs, *d1* and *d2*, as parameters and returns the amount of days between *d1* and *d2*. For example, there are two days between May 30, 2016 and June 1, 2016.

NB: If *d2* comes before *d1*, your function should return the negative of the amount of days between *d2* and *d1*.You may use the *compare* function from Part (b) and the *nextDay* function from Part (c). *[6 marks]*

Total Marks for Question 2 is 20

Basics

Sunday, December 2, 2018 3:04 PM



INIT

Example: Init all Matrix entries to zero

```
9   void zeros(int A[][MAX_COLS], int n, int m){
10      for(int i = 0; i < n; i++){
11          for(int j = 0; j < m; j++){
12              A[i][j] = 0;
13          }
14      }
15  }
```

TRAVERSING

Nested loop Concept:
i from 0 to n-1
j from 0 to m-1.

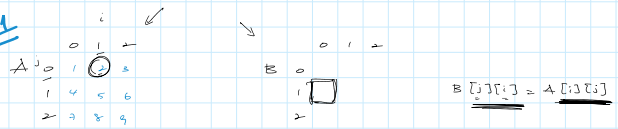
```
18  void print(int A[][MAX_COLS], int n, int m){
19      for(int i = 0; i < n; i++){
20          for(int j = 0; j < m; j++){
21              cout << A[i][j] << "\t";
22          }
23          cout << endl;
24      }
25      cout << endl;
26  }
```

1) Write a function which given a 2D array A representing 3 x 3 matrix, and the number of rows and columns in A, transposes A. The function also accepts a matrix B which holds the transpose of A.

The transpose of a matrix is a new matrix which is formed by turning all the rows of a given matrix into columns and vice-versa.

For example given a matrix A: $\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\\7 & 8 & 9\end{bmatrix}$ the transpose of A is $\begin{bmatrix}1 & 4 & 7\\2 & 5 & 8\\3 & 6 & 9\end{bmatrix}$

VERSION 1




```
29  void transpose1(int A[][MAX_COLS], int B[][MAX_COLS], int n, int m){
30      for(int i = 0; i < n; i++)
31          for(int j = 0; j < m; j++)
32              B[j][i] = A[i][j];
33  }
34
35
36  void transpose2(int A[][MAX_COLS], int n, int m){
37      for(int i = 0; i < n; i++)
38          for(int j = i+1; j < m; j++)
39              int temp = A[i][j];
40              A[i][j] = A[j][i];
41              A[j][i] = temp;
42  }
```

VERSION 2

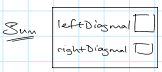
2) Write a function which given a 2D array A representing 3 x 3 matrix, and the number of rows and columns in A, returns true if the matrix is a diagonal matrix.

A diagonal matrix is a square matrix which has zeroes everywhere except the main diagonal.

For example: $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 9 \end{bmatrix}$ is a diagonal matrix.

```
45  bool isDiagonal(int A[][MAX_COLS], int n, int m){
46      for(int i = 0; i < n; i++)
47          for(int j = i+1; j < m; j++)
48              if(!(A[i][j] == 0 && A[j][i] == 0))
49                  return false;
50      return true;
51  }
```

3) Write a function to find the sum of the diagonal elements of a 3 x 3 matrix as shown in the figure. There are two diagonals in a square matrix the left diagonal and the right diagonal. Your function should return both sums.




```
struct Sum {
    int l, r;
};
```

```
71  Sum sumDiagonals2(int A[][MAX_COLS], int n, int m){
72      Sum s;
73      s.left = 0;
74      s.right = 0;
75      for(int i = 0, j = m-1; i < n, j >= 0; i++, j--){
76          s.left = s.left + A[i][i];
77          s.right = s.right + A[i][j];
78      }
79      return s;
80  }
```

4) Write a function which given a 2D array A representing 3 x 3 matrix, and the number of rows in A interchanges the diagonal elements of a 3 x 3 matrix as shown in the figure below.

5) Write a function which given a 2D array A representing 3 x 3 matrix, and the number of rows in A returns true id A is a lower triangular matrix and false otherwise.

A square matrix is called lower triangular if all the entries above the main diagonal are zero.

For example: $\begin{bmatrix} 1 & 0 & 0 \\ 2 & 5 & 0 \\ 6 & 8 & 9 \end{bmatrix}$ is a lower triangular matrix.

6) Write a function which given a 2D array A representing 3 x 3 matrix, and the number of rows in A returns the sum of the elements in an upper triangular matrix.

A square matrix is called upper triangular if all the entries below the main diagonal are zero. We need to find the sum of the numbers enclosed in red

(a) An $m \times n$ matrix is considered *sparse* if *at most* 20% of its elements are non-zero. A matrix `MM` is declared as `int MM[200][200]`. Write programming code to read, from a file *input.txt*, values for m and n, followed by mn integers to fill the matrix in *row-order*. Assume $m, n \leq 200$. Your code then calls the function *isSparse* (see next) and prints a message indicating whether or not the matrix is sparse. Include all relevant declarations.

Write a function *isSparse* which, given a matrix like `MM` and values for m and n, returns 1 if the matrix is sparse and 0, otherwise. **[4]**

b. A *square* matrix is a matrix where the number of rows is equal to the number of columns. M is a square matrix of size n. A square matrix is *sparse* if it contains less than 20% of non-zero values. Write a function which returns *true* if M is sparse and *false* otherwise. The prototype for sparse is:

```
bool isSparse(int M[][n], int n);
```

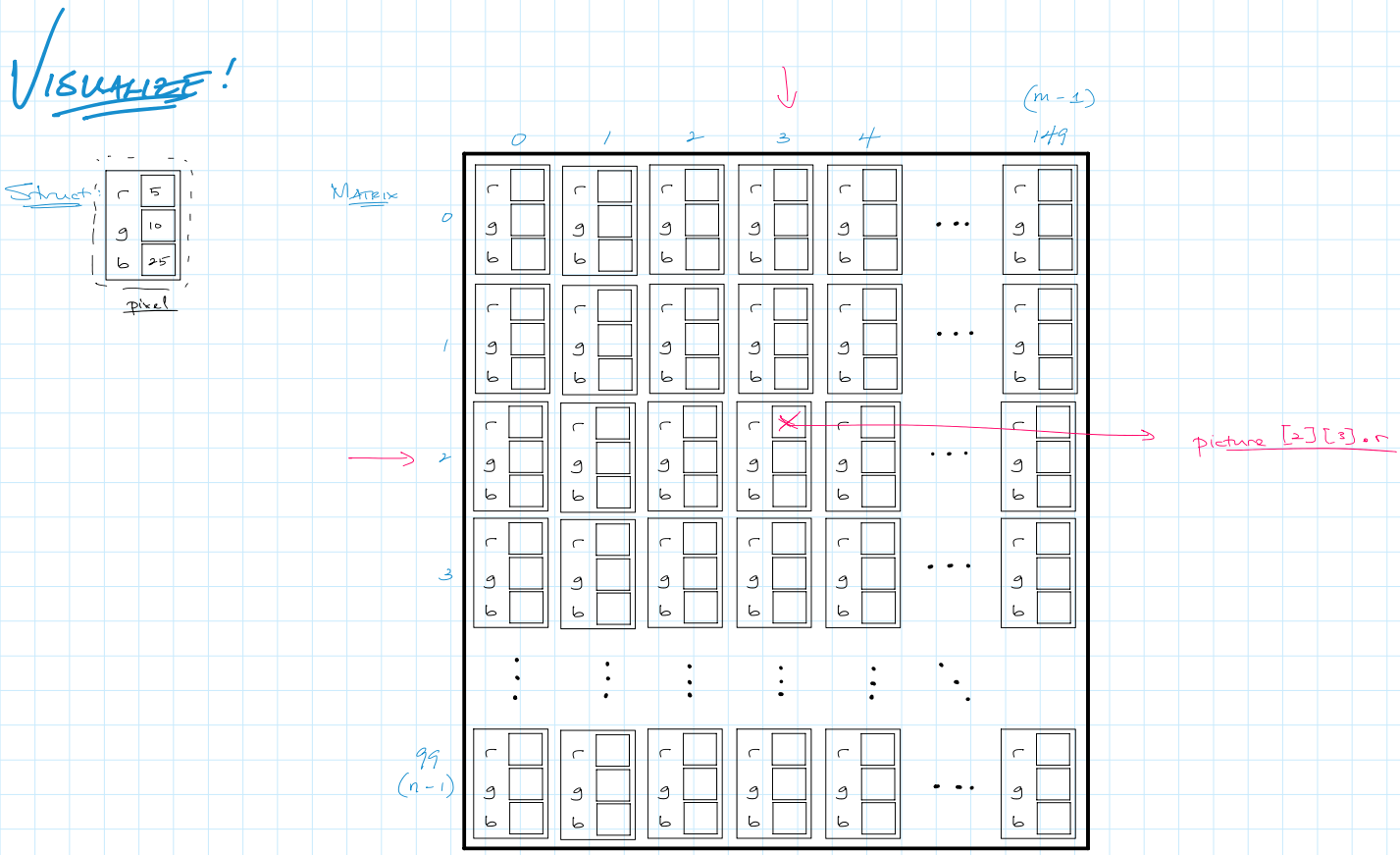
[4]

Total marks: 10

```
92  bool isSparse(int A[][MAX_COLS], int n){
93      int nonZeroCount = 0;
94      for(int i = 0; i < n; i++)
95          for(int j = 0; j < n; j++)
96              if(A[i][j] != 0)
97                  nonZeroCount++;
98
99      return nonZeroCount < 0.2*n*n;
100 }
```

2. A certain image, `picture`, consists of a set of pixels (picture elements) stored in a two-dimensional array of `100` rows by `150` columns.

Each pixel has a red value, a green value and a blue value which gives the pixel its colour. All three values are integers in the range 0 to 255.



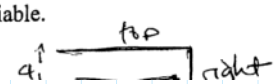
a. Declare a struct of type `Pixel` to store a pixel's colour data. [1]

```
3   struct Pixel{
4       int r, g, b;
5   };
```

b. Write a function, `colour`, which accepts a `Pixel` variable and the red, green and blue values of the pixel as parameters and returns a `Pixel` variable with the given values. [2]

```
8   Pixel color(Pixel pixel, int r, int g, int b){
9       pixel.r = r;
10      pixel.g = g;
11      pixel.b = b;
12      return pixel;
13  }
```

c. Declare a `Pixel` variable, `p` and assign the colour *orchid* to this variable. (orchid: red = 218, green = 112, blue = 214) [2]

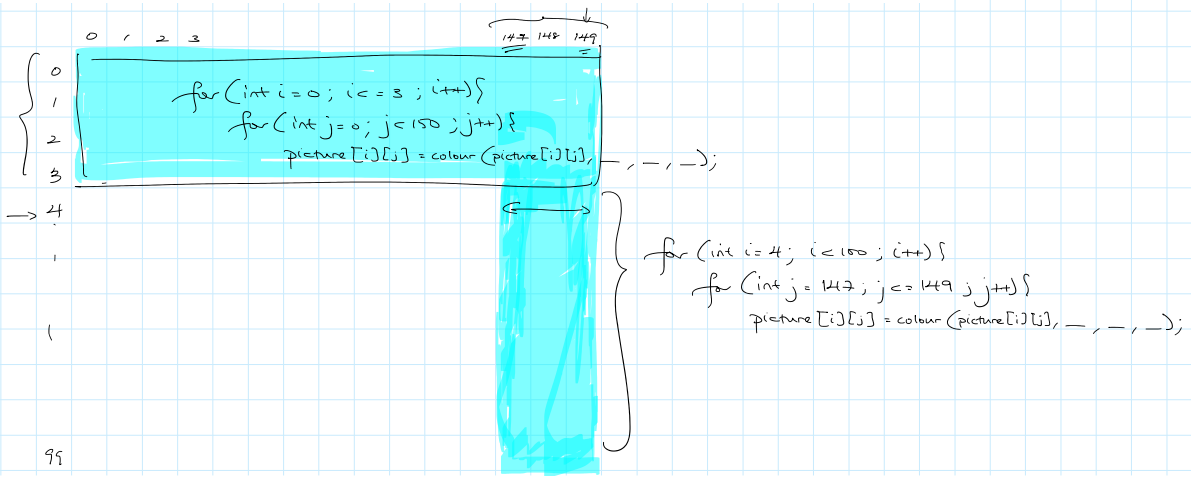



```
16  Pixel p;
17  p = color(p, 218, 112, 214);
```

d. Declare `picture` as a two dimensional array of `Pixel` variables. [1]


```
20  Pixels picture[100][150];
```

e. Write a segment of code to colour all the pixels on the **top** and **right** edges of the image orchid. The top edge has a thickness of 4 pixels and the right edge has a thickness of 3 pixels. So, you must colour the four rows of pixels at the extreme top and the three rows of pixels at the extreme right of the image in orchid. [4]



f. Assume that the data for an image has already been read from a file and stored in `picture`.

Write code to change the `Pixel` variables in `picture` to grayscale using the following procedure:

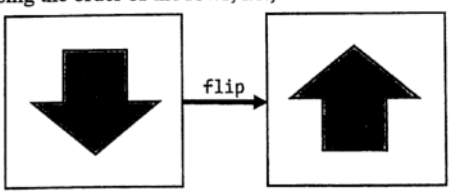
- For each pixel, use 30% of its red value, 59% of its green value and 11% of its blue value and set each colour of the pixel to this calculated value. [5]

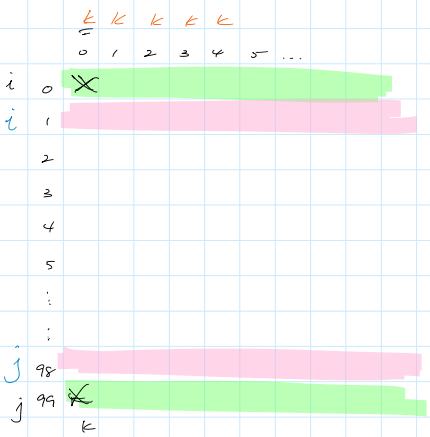
```
23  for(int i = 0; i < 100; i++){
24      for(int j = 0; j < 150; j++){
25          int newColour = 0.3*picture[i][j].r + 0.59*picture[i][j].g + 0.11*picture[i][j].b;
26          picture[i][j] = color(picture[i][j], newColour, newColour, newColour);
27      }
28  }
```

g. The image in `picture` can be *flipped* by reversing the order of the rows, i.e.,

- Row 0 is switched with row 99
- Row 1 is switched with row 98
- Row 2 is switched with row 97
- And so on.

Write code to flip the image in `picture`. [5]





```
31  int i = 0;
32  int j = 99;
33  while(i < j){
34      for(int k = 0; k < 150; k++){
35          Pixel temp = picture[i][k];
36          picture[i][k] = picture[j][k];
37          picture[j][k] = temp;
38      }
39      i++;
40      j--;
41  }
```

half or >

h. If at least half of the pixels in an image is coloured black (red = 0, green = 0, blue = 0), it is referred to as a *sparse* image. Write code to determine if `picture` is sparse. [5]

Total marks: 25

Board Game***

Wednesday, December 5, 2018 2:58 AM

17-18 S2 COMP1602 Final

2) A game is played on a board which is stored in a two dimensional (2D) array. The 2D array to represent the game board has a size of *numRows* and *numCols* where `0 ≤ numRows < 100`, `0 ≤ numCols < 100`, and *numRows* = *numCols*. A *Cell* struct stores the following data for **each** cell in the 2D array:

- Does the cell contain a light bulb?
- Is the cell hidden?
- The amount of immediate neighbours that contain a light bulb.

(a) Declare a struct, *Cell,* to store information on each cell and declare the array, *board*, to store the game board.

[2 marks]

(b) Write a function, *initBoard*, which initializes the array such that each cell:

- Does not contain a light bulb
- Is hidden
- Has zero neighbours which contain light bulbs

The prototype of *initBoard* is as follows:

```
void initBoard (Cell board[][100], int numRows);
```

[4 marks]

(c) Write a function, *isValid*, which given a particular location (row, column), determines if the location is valid for the given board. A location is valid if it is within the range of the rows and columns of the initialized board. The prototype of *isValid* is as follows:

```
bool isValid (Cell board[][100], int numRows, int row, int col);
```

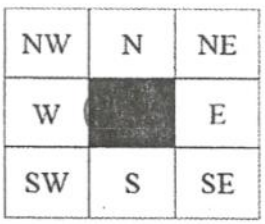
[3 marks]

(d) Write a function, *placeBulbs*, to randomly assign 10 percent of all the locations in the board with light bulbs. Note that every location on the board has two components, a row number and a column number. The prototype of *placeBulbs* is as follows:

```
void placeBulbs (Cell board[][100], int numRows);
```

[5 marks]

(e) A neighbour of a cell (shown in black on the diagram) is a cell that is North (N), South (S), East (E), West (W), North East (NE), South East (SE), South West (SW) or North West (NW) of the given cell, as shown in the diagram.



One or more neighbours may not exist if the cell in on the edge of the board.

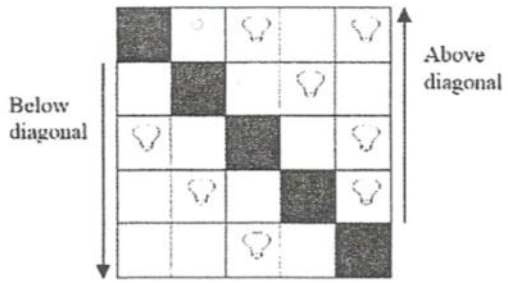
Write a function, *getNumber*, which given the 2D board and the location of a particular cell, finds the amount of neighbours of the cell which contain a light bulb. You can use functions previously written.

The prototype of *getNumber* is as follows:

```
int getNumber (Cell board[][100], int numRows, int row, int col);
```

[6 marks]

(f) The game board is said to be *crooked* if there are more light bulbs in the cells above the main diagonal than in the cells below the main diagonal. For example, the following game board is crooked:



Write a function, *isCrooked*, which determines if the game board is crooked. The prototype of *isCrooked* is as follows

```
bool isCrooked (Cell board[][100], int numRows);
```

[5 marks]

Total Marks: 25

Question***

Wednesday, December 5, 2018 3:29 AM

COMP1404_2_16

3. Figure 1 shows a 2-dimensional array of integers, *m*:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4	4	4	4	4	4	4	4

Figure 1

(a) Write code to declare the array, *m*. *[1 mark]*

(b) The array shown in Figure 1 has already been loaded with values. Write a segment of code that was used to assign the values shown in Figure 1 to the array. *[3 marks]*

(c) Write a segment of code to interchange the row containing the 2's with the row containing the 4's. *[4 marks]*

(d) Write a segment of code to randomly generate a location in the array and assign the value -1 to that location in the array. *[2 marks]*

Total Marks for Question 3 is 10

Distinct***

Wednesday, December 5, 2018 3:42 AM

COMP1404_1_14

c) A 5 x 5 array ***arr***, is filled to capacity with positive integers ranging from 1 to 65. Write code to print the number of distinct values contained in ***arr***. [5]

Sums, row and column sum

Wednesday, December 5, 2018 3:49 AM

16-17 S3 COMP1602 Final

2. **(a)** **A 4 X 5 array, arr, is filled to capacity with positive integers. Write code to find and print:**

(i) **the sum of the odd integers in the array.**

```
int OddRow[4] = {0}
int OddCol[5] = {0}
```

(ii) **the largest column sum. The column sum is defined as the sum of the elements in a particular column.**

(iii) **the column number that has the largest column sum.**

[10]

C String Notes

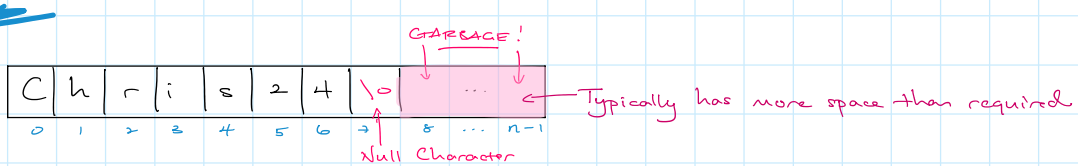
Sunday, December 2, 2018 3:03 PM

The ASCII table [Characters Mapped to Integers] → C++. Show DEMO

Dec	Hex	Oct	Chr	Dec	Hex	Oct	HTML	Chr	Dec	Hex	Oct	HTML	Chr	Dec	Hex	Oct	HTML	Chr
0	0	000	NULL	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	Start of Header	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	Start of Text	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	End of Text	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	End of Transmission	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	Enquiry	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	Acknowledgment	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	Bell	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	Backspace	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	Horizontal Tab	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	Line feed	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	Vertical Tab	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	&#10	k
12	C	014	Form feed	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	&#10	l
13	D	015	Carriage return	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	&#109	m
14	E	016	Shift Out	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	&#110	n
15	F	017	Shift In	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	&#111	o
16	10	020	Data Link Escape	48	30	060	0	0	80	50	120	P	P	112	70	160	&#112	p
17	11	021	Device Control 1	49	31	061	1	1	81	51	121	Q	Q	113	71	161	&#113	q
18	12	022	Device Control 2	50	32	062	2	2	82	52	122	R	R	114	72	162	&#114	r
19	13	023	Device Control 3	51	33	063	3	3	83	53	123	S	S	115	73	163	&#115	s
20	14	024	Device Control 4	52	34	064	4	4	84	54	124	T	T	116	74	164	&#116	t
21	15	025	Negative Ack.	53	35	065	5	5	85	55	125	U	U	117	75	165	&#117	u
22	16	026	Synchronous idle	54	36	066	6	6	86	56	126	V	V	118	76	166	&#118	v
23	17	027	End of Trans. Block	55	37	067	7	7	87	57	127	W	W	119	77	167	&#119	w
24	18	030	Cancel	56	38	070	8	8	88	58	130	X	X	120	78	170	&#120	x
25	19	031	End of Medium	57	39	071	9	9	89	59	131	Y	Y	121	79	171	&#121	y
26	1A	032	Substitute	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	&#122	z
27	1B	033	Escape	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	File Separator	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	Group Separator	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	Record Separator	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	Unit Separator	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	Del

asciicharstable.com

VISUALIZE



C-Strings are Special Arrays

They behave differently from numerical arrays.

Just accept it !!

Differences Example:

(1) Null

(2) Input

(3) Output

isLetter

Wednesday, December 5, 2018 4:00 AM

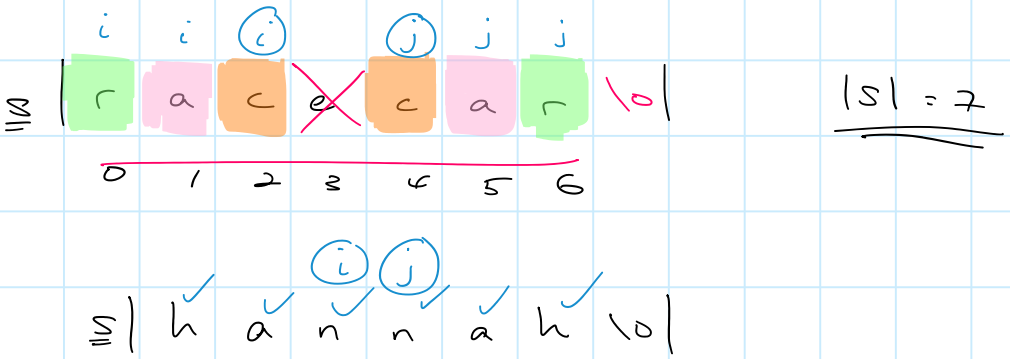
isDigit

Wednesday, December 5, 2018 4:00 AM

isPalindrome

Wednesday, December 5, 2018 3:59 AM

Write a boolean function to determine whether a cstring is a Palindrome




```
87  bool isPalindrome(char s[]){
88      int i = 0;
89      int j = strlen(s)-1;
90      while(i < j){
91          if(s[i] != s[j])
92              return false;
93          i++;
94          j--;
95      }
96      return true;
97  }
```

Reverse

Wednesday, December 5, 2018 4:01 AM

Write a function to reverse a cstring.

Strcat***

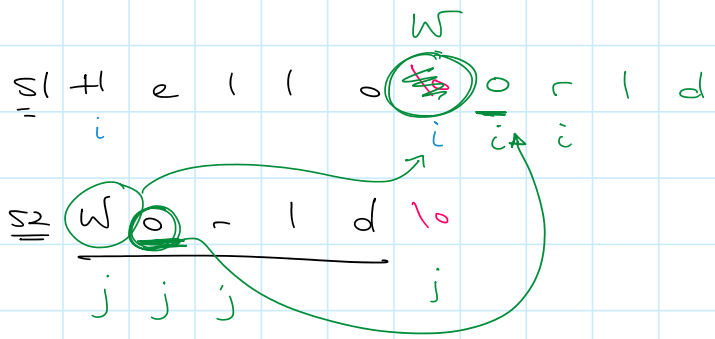
Wednesday, December 5, 2018 2:46 AM

17-18 S2 COMP1602 Final

(a) Write a function, *append*, which accepts two C-strings ***s*** and ***t*** as parameters and adds the contents of ***t*** to the *end* of the contents of ***s***. Assume that ***s*** has enough space to hold the contents of ***t***. The prototype of *append* is as follows:

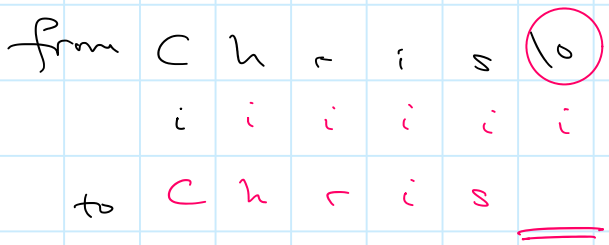
```
void append (char s[], char t[]);
```

[3 marks]



Strcpy

Friday, December 7, 2018 12:29 PM




```
51  void strcpy(char to[], char from[]){
52      int i = 0;
53      while(s[i] != '\0'){
54          to[i] = from[i];
55      }
56      to[i] = '\0';
57  }
```

GetInt***

Wednesday, December 5, 2018 2:47 AM

17-18 S2 COMP1602 Final
comp1404_2_14

(b) A C-string ***s*** contains a single integer value embedded among other characters, e.g., `"weight = 115 kg".` Write a function, *getInt*, which accepts ***s*** as a parameter and returns the integer *value* stored in ***s***. Given the example C-string, *getInt* should return 115.

[6 marks]

(b) A character array **s** contains an embedded integer number, for example "weight =34kg". Write a function with prototype **int extractInt (char s[],int size)** which returns the integer value embedded in **s** where size is the maximum length that **s** can be. [3]

Panagram***

Wednesday, December 5, 2018 2:52 AM

17-18 S2 COMP1600 Final

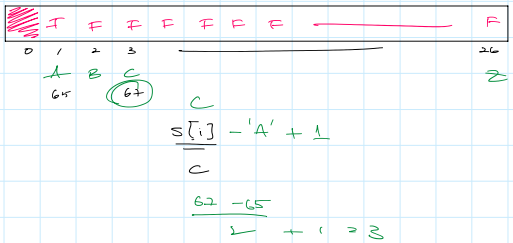
(c) A *pangram* is a series of words which contains all the letters of the alphabet. The letters of the words may be in upper case or lower case. The most famous English pangram is probably:

s = "The Quick Brown Fox Jumps Over The Lazy Dog"

Write a function, *isPangram*, which accepts a C-string *s* as a parameter and returns *true* if the words in *s* form a pangram and *false* otherwise.

[6 marks]

IDEA:




```
21  bool isPanagram(char s[]){
22      int i = 0;
23      bool alphabet[27] = {false};
24      while(s[i] != '\0'){
25          if(isLetter(s[i])){
26              int j = position(s[i]);
27              // cout << s[i] << '\t' << j << endl;
28              alphabet[j] = true;
29          }
30          i++;
31      }
32
33      for(int i = 1; i <= 26; i++){
34          // cout << i << '\t' << alphabet[i] << endl;
35          if(alphabet[i] == false)
36              return false;
37      }
38      return true;
39
40  }
```

Longest Sequence***

Wednesday, December 5, 2018 2:58 AM

17-18 S2 COMP1602 Final

(d) A C-string ***s*** contains a sequence of lowercase letters of the alphabet. Write a function, *longestSequence*, which accepts ***s*** as a parameter and returns the *length* of the *longest sequence* of *identical letters* in ***s***. For example, the function should return 6 if ***s*** is "aabbcddddddeeebbbaaaacccd".

[5 marks]

Strlen***

Wednesday, December 5, 2018 3:03 AM

16-17 S2 COMP1602 Final

1. A C-string is a character array with a null terminating character ('\0'). Write code to implement the following functions involving C-strings. You are *not* allowed to use any of the functions built into the C-string library.

a. `int length(char s[])`

/* Returns the number of characters in `s`. */ [2]

```
43  int strlen(char s[]){
44      int i = 0;
45      while(s[i] != '\0')
46          i++;
47      return i;
48  }
```

Add***

Wednesday, December 5, 2018 3:05 AM

16-17 S2 COMP1602 Final

b. `int add(char s[], char c)`

/* Inserts a character `c` at the end of `s` and returns the length of the C-string after the insertion.*/ [2]

Last Occur***

Wednesday, December 5, 2018 3:05 AM

16-17 S2 COMP1602 Final

c. `int lastOccur (char s[], char c)`

/* Searches for and returns the location of the last occurrence of the character `c` in `s`.

If `c` is not in `s` return -1.*/ [2]

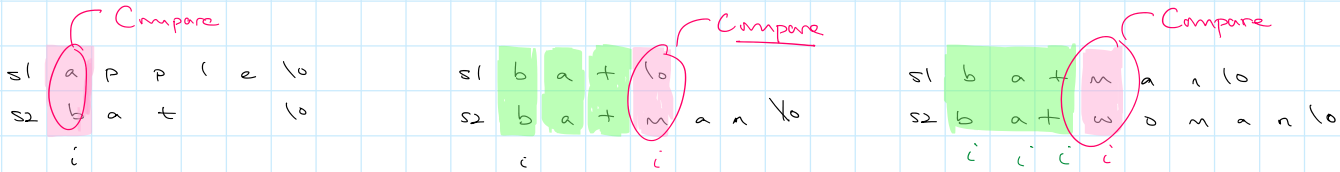
Strcmp***

Wednesday, December 5, 2018 3:06 AM

16-17 S2 COMP1602 Final

d. `int compareTo (char s1[], char s2[])`

/* Returns 0 if **s1** and **s2** are the same; -1 if **s1** < **s2** and 1 if **s1** > **s2**. */ [5]




```
76    int strcmp(char s1[], char s2[]){
77        int i = 0;
78        while(s1[i] != '\0' && s2[i] != '\0' && s1[i] == s2[i])
79            i++;
80
81        if(s1[i] < s2[i]) return -1;
82        else if(s1[i] == s2[i]) return 0;
83        else return 1;
84    }
```

Haystack***

Wednesday, December 5, 2018 3:06 AM

16-17 S2 COMP1602 Final

e. A C-string, `haystack`, contains a special message. The message is enclosed by the tags '<' and '>'. Write a function which takes two C-strings, `haystack` and `message`, as parameters and finds and stores the message in `message`. The function has the following prototype:

```
void getMessage (char haystack[], char message[])
```

[4]

ShiftY***

Wednesday, December 5, 2018 3:19 AM

COMP1404_2_16

1. The Shift*Y* algorithm can be used to encrypt a string (character **array** terminated with the null character) before it is transmitted over the Internet. The Shift*Y* algorithm works as follows:

- Each letter in the message is converted to the letter that is *Y* positions to the right of that letter in the English alphabet. The case of the letters must be maintained.

  e.g., Using Shift*4* (*Y* is 4): 'a' → 'e', 'B' → 'F', 'w' → 'a' and 'Z' → 'D'.
  Also, using Shift*5* (*Y* is 5): 'a' → 'f', 'B' → 'G', 'w' → 'b' and 'Z' → 'E'.

- Non-letters are left as they are.

(a) What would the following string be encrypted to by the Shift*4* algorithm?

"Madam, I'm Adam!" *[2 marks]*

(b) Write a function *encrypt* which encrypts a letter of the alphabet, given the value of *Y* which is supplied as a parameter. The function should return the encrypted letter. The prototype of *encrypt* is as follows:

```
char encrypt(char ch, int y);
```

e.g., encrypt('a', 4) should return 'e' and encrypt('B', 5) should return 'G'. *[5 marks]*

(c) Write a function *decrypt* which returns a letter to its original form, given the value of *Y* which is supplied as a parameter. The function should return the original letter. The prototype of *decrypt* is as follows:

```
char decrypt (char ch, int y);
```

e.g., decrypt('e', 4) should return 'a' and decrypt('E', 5) should return 'Z'. *[3 marks]*

(d) Write a function *shiftY* which accepts a string containing the message to be encrypted, a string to hold the encrypted message, and the value of *Y* as parameters and encrypts the string using the Shift*Y* algorithm. The prototype of *shiftY* is as follows:

```
void shiftY(char message[], char encryptedMessage[], int y);
```

[5 marks]

(e) A text file, `message.txt`, contains a message to be encrypted (no more than 1000 characters). Write a segment of code to read the contents of the file into a string and then encrypt the string using the *shiftY* function from Part (d). Assume that *Y* is 4. *[5 marks]*

Total Marks for Question 1 is 20

Passage.txt and getWord***

Wednesday, December 5, 2018 3:34 AM

comp1404_2_14

1.

(a) A program is required to process the text file "**passage.txt**" and output an alphabetical listing of all the different words in an English passage together with the number of times the word appear. The listing is to be printed in **descending** order by frequency of appearance. One or more blanks, a period, a comma, a semi-colon, a question mark, or an exclamation mark delimit words. **No word is longer than 20 characters**.

(i) Declare a struct of type **WordInfo** to store a word and its frequency. [1]

(ii) Write a function **isLetter** that accepts a character argument and return **1** if it is a letter and **0** otherwise. [2]

(iii) Write a function **toLower** that accepts a character array that contains a word and converts it to all lowercase [2]

(iv) Write a function with the prototype **int getWord (FILE * in, char word[])** that extracts a word from **passage.txt** and returns 1 if there are no more words to extract and 0 otherwise. You must read from the file character by character. [6]

(v) Write a main function that uses the functions and declaration above to complete the program as outlined. **You may assume there are no more than 100 distinct words**. [6]

indexOf***

Wednesday, December 5, 2018 3:40 AM

COMP1404_1_14

Question 2 **[Total 15 marks]**

a) Write a method **int indexOf(char s1[], char s2[], int m, int n)**, which takes two string arguments ***s1*** and ***s2*** and two integer arguments ***m*** and ***n*** (where ***m*** is the number of characters in ***s1*** and ***n*** is the number of characters in ***s2***). This function returns the starting index of the first occurrence of ***s1*** in ***s2*** if ***s1*** is a substring of ***s2***; it returns **-1** if ***s1*** is not a substring of ***s2***. You are NOT allowed to use the **string.h** file.

[5]

No Comments***

Wednesday, December 5, 2018 3:43 AM

COMP1404_1_14

a) Write a program, **decomment.c**, which processes another program, **assignment.c**, by removing all the ***single-line comments*** from **assignment.c** (that is comments which begin with the character pair //). Output should be written to the file ***assignmentWithNoComments.c***. [5]

Planning

Wednesday, December 5, 2018 1:43 AM

Alternate concept
Just leave space In the notes to work things out

Day	Original Plan	Reality	Altered Plan
1	Searching, Sorting, Random numbers	Searching, Selection Sort (Slow/Medium Pace in 3.5 hrs)	
2	Structs, Matrices	No class	Insertion Sort, Random Numbers, Structs (Abandoned topics from day 1)
3	C-Strings	Matrices, C-Strings (Medium Pace) (Success)	Matrices, C-Strings

Chad Flyer

Sunday, December 2, 2018 1:36 PM

Need help with programming?

Crash Course Registration
NOW OPEN!!!

Final exam preparation for Programming 2 ((5th, 6th, 7th December, 9am to 12pm))

and Data Structures ((10th, 11th, 12th, 13th December, 9am to 12pm))

Location (10 minutes from UWI):
53, Kassie Street, El Dorado (Mr Ali's Physics Lessons)
https://goo.gl/maps/ecMZ1sy8mM72

Will cover theory, code, examples and past finals questions for all topics.

At a *very* student friendly cost

Limited spaces available

See link for more info