Lab 09

Command Line Arguments

- Generally we ask user for input using standard I/O
- We can directly pass information into our program using the command line terminal

Passing information inside main() function

```
//old main

int main ()

{

//stuff

}
```

Passing information using the command line

```
//new main
int main( int argc, char**
argv )
{
   //stuff
}
```

Command Line Arguments

argc: a count of how many command line arguments (including the program's name) were passed in

argv: A 2D character array with all the words passed in

Converting Strings

- There are functions that can convert string type to another type (for eg. an **int** or **double)**
- **stoi ( ) //converts a string type to an integer type**
- **stod ( ) //converts a string type to a double type**

Passing Arrays as arguments to functions

- There are two ways to that. First you can actually pass a pointer to the array as in:

```
void printArray (int * arr, int size) //passing a pointer to an array allocated on the heap

{

}

Void printArray (int arr[] , int size) {

}
```

Passing 2D Arrays

Two syntaxes:

type ** parameter_name

Or

type* parameter_name[]

Exercise

int* insert(int arr[ ], int& size, int value, int position)

- Inserts the given value at the specified position
- Creates a new array, copies all old value over adjusting indices as necessary
- Deletes the old array (arr)
- Updates the size (that's why it's passed by reference)
- Returns a pointer to the new array

Exercise

int* remove(int arr[ ], int& size, int position)

- Removes the value at the given position
- Creates a new array, copies all old value over adjusting indices as necessary
- Deletes the old array (arr)
- Updates the size (that's why it's passed by reference)
- Returns a pointer to the new array

Exercise

int count(int arr[ ], int size, int target)

- returns a count of how many times the target value is in the array

void print(int arr[ ], int size)

- Prints array as required