Course: ENSF 614 - Fall 2023

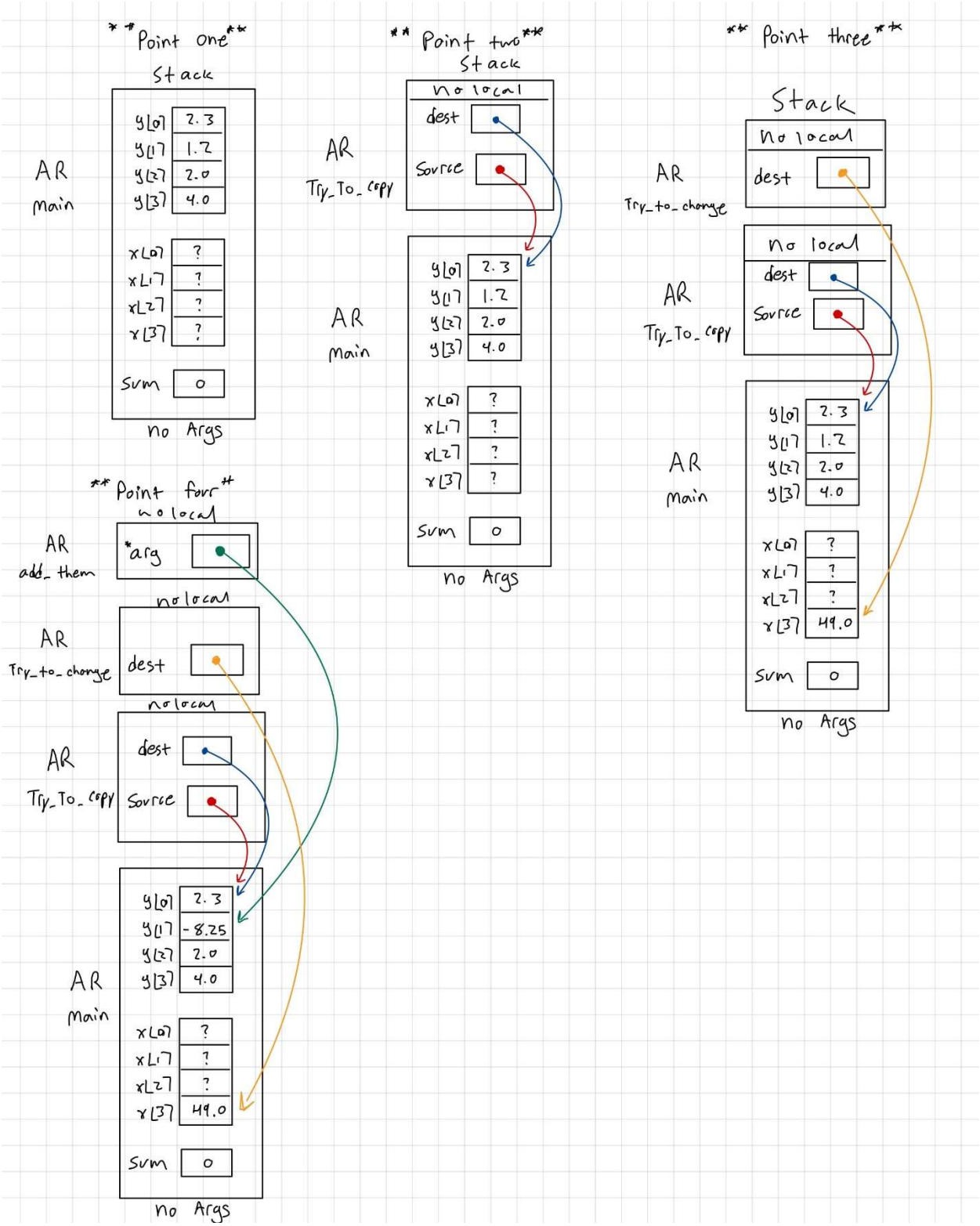
Lab #: Lab 2

Instructor: Prof. M. Moussavi

Student Name: Jeremy Sugimoto

Submission Date: September 27, 2023

Exercise A:


** Point one **
Stack
AR
main
y[0] 2.3
y[1] 1.7
y[2] 2.0
y[3] 4.0
x[0] ?
x[1] ?
x[2] ?
x[3] ?
sum 0
no Args
** Point two **
Stack
AR
Try_To_copy
no local
dest
Source
AR
main
y[0] 2.3
y[1] 1.7
y[2] 2.0
y[3] 4.0
x[0] ?
x[1] ?
x[2] ?
x[3] ?
sum 0
no Args
** Point three **
Stack
AR
Try_to_change
no local
dest
AR
Try_To_copy
no local
dest
Source
AR
main
y[0] 2.3
y[1] 1.7
y[2] 2.0
y[3] 4.0
x[0] ?
x[1] ?
x[2] ?
x[3] 49.0
sum 0
no Args
** Point four **
no local
AR
add_them
*arg
no local
AR
Try_to_change
dest
no local
AR
Try_To_copy
dest
Source
AR
main
y[0] 2.3
y[1] -8.25
y[2] 2.0
y[3] 4.0
x[0] ?
x[1] ?
x[2] ?
x[3] 49.0
sum 0
no Args

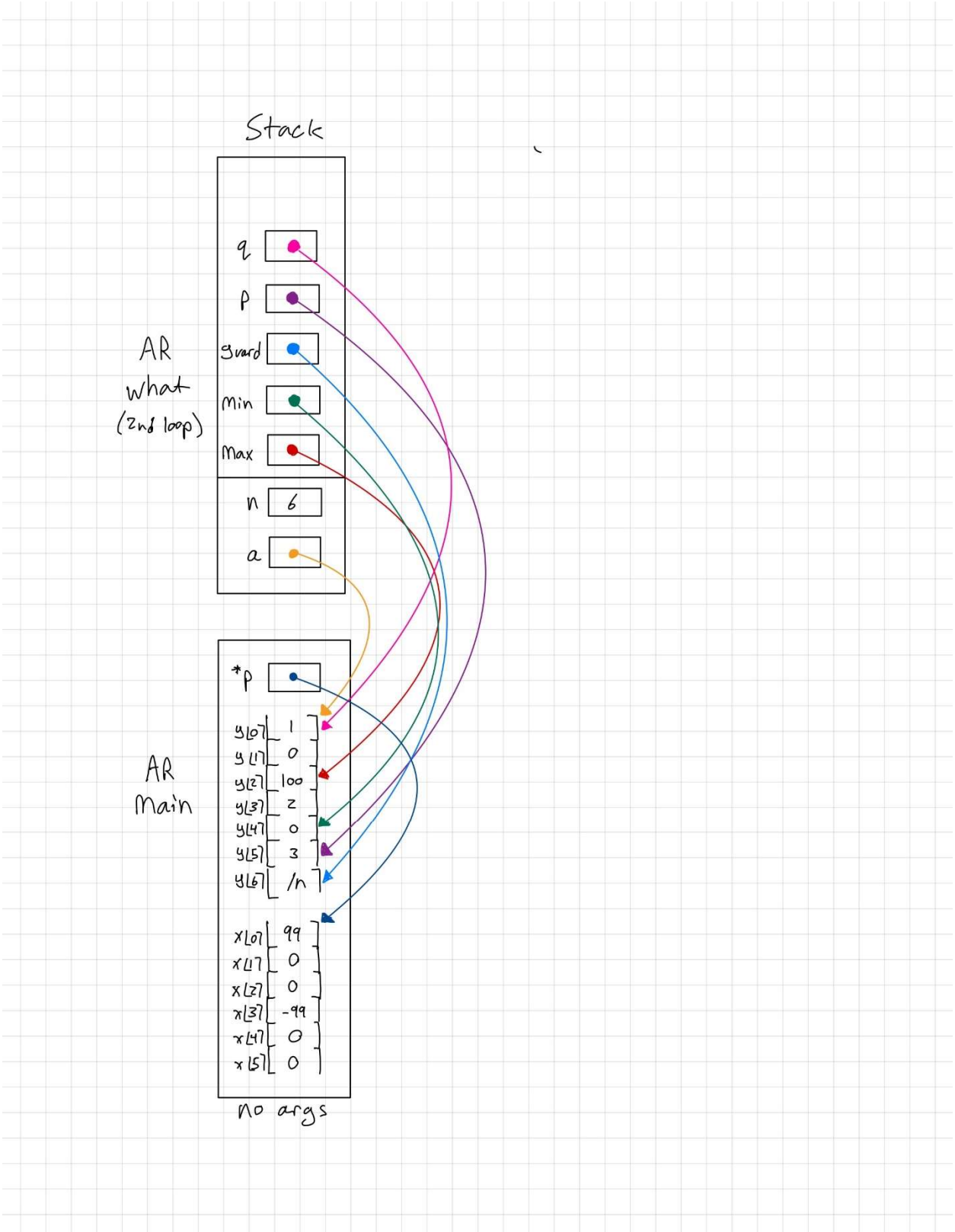
Exercise B:

```
/*
 *  lab2exe_B.cpp
 *  ENSF 614 Lab 2 Exercise B
 *  Completed by: Jeremy Sugimoto
 *  Submission Date: Sept 27, 2023
 */


#include <iostream>
#include <cstring>
using namespace std;


int my_strlen(const char *s){

int i = 0;
while (s[i]){
    i++;
}
return i;
}

void my_strncat(char *dest, const char *source, int n){
    int dest_len = 0;

    // Calculate the length of the destination string
    while (dest[dest_len] != '\0') {
        dest_len++;
    }

    // Copy characters from source to dest, up to n characters
    for (int i = 0; i < n && source[i] != '\0'; i++) {
        dest[dest_len + i] = source[i];
    }

    // Ensure null-termination of dest
    dest[dest_len + n] = '\0';
}

int my_strcmp(char* str1, char* str2){
    int diff = 0;
    int len = 0;
    while (str1[len] != '\0' && str2[len] != '\0') {
        len++;
```

```
    }
    for (int i = 0; i< len; i++){
        if (str1[i] != str2[i]){
            diff = int(str1[i]) - int(str2[i]);
            break;
        }
    }
    return diff;
}
int main(void)
{
    char str1[7] = "banana";
    const char str2[] = "-tacit";
    const char* str3 = "-toe";

    /* point 1 */
    char str5[] = "ticket";
    char my_string[100]="";
    int bytes;
    int length;

    /* using strlen libarary function */
    length = (int) my_strlen(my_string);
    cout << "\nLine 1: my_string length is " << length;

    /* using sizeof operator */
    bytes = sizeof (my_string);
    cout << "\nLine 2: my_string size is " << bytes << " bytes.";

    /* using strcpy libarary function */
    strcpy(my_string, str1);
    cout << "\nLine 3: my_string contains: " << my_string;

    length = (int) my_strlen(my_string);
    cout << "\nLine 4: my_string length is " << length << ".";

    my_string[0] = '\0';
    cout << "\nLine 5: my_string contains:\"" << my_string << "\"";

    length = (int) my_strlen(my_string);
    cout << "\nLine 6: my_string length is " <<  length << ".";

    bytes = sizeof (my_string);
    cout << "\nLine 7: my_string size is still " << bytes << " bytes.";
```

```
    /* strncat append the first 3 characters of str5 to the end of my_string */
    my_strncat(my_string, str5, 3);
    cout << "\nLine 8: my_string contains:\"" << my_string << "\"";

    length = (int) my_strlen(my_string);
    cout << "\nLine 9: my_string length is " << length << ".";

    my_strncat(my_string, str2,  4);
    cout << "\nLine 10: my_string contains:\"" << my_string << "\"";

    /* strncat append ONLY up ot '\0' character from str3 -- not 6 characters */
    my_strncat(my_string, str3, 6);
    cout << "\nLine 11: my_string contains:\"" << my_string << "\"";

    length = (int) my_strlen(my_string);
    cout << "\nLine 12; my_string has " << length << " characters.";

    cout << "\n\nUsing strcmp - C library function: ";

    cout << "\n\"ABCD\" is less than \"ABCDE\" ... strcmp returns: " <<
    my_strcmp("ABCD", "ABCDE");

    cout << "\n\"ABCD\" is less than \"ABND\" ... strcmp returns: " <<
    my_strcmp("ABCD", "ABND");

    cout << "\n\"ABCD\" is equal than \"ABCD\" ... strcmp returns: " <<
    my_strcmp("ABCD", "ABCD");

    cout << "\n\"ABCD\" is less than \"ABCd\" ... strcmp returns: " <<
    my_strcmp("ABCD", "ABCd");

    cout << "\n\"Orange\" is greater than \"Apple\" ... strcmp returns: " <<
    my_strcmp("Orange", "Apple") << endl;
    return 0;
}
```

Exercise C:


Stack
AR
what
(2nd loop)
q
p
guard
min
max
n
6
a
AR
main
*p
y[0]
1
y[1]
0
y[2]
100
y[3]
2
y[4]
0
y[5]
3
y[6]
/n
x[0]
99
x[1]
0
x[2]
0
x[3]
-99
x[4]
0
x[5]
0
no args

Exercise E:

```
/*
 * lab2exe_E.cpp
 * Implementation file for complex number module
 *
 * ENSF 614 Lab 2 Exercise E
 * Completed by: Jeremy Sugimoto
 * Submission Date: Sept 27, 2023
 */

#include "lab2exe_E.h"

cplx cplx_add(cplx z1, cplx z2)
{
  cplx result;

  result.real = z1.real + z2.real;
  result.imag = z1.imag + z2.imag;
  return result;
}
void cplx_subtract(cplx z1, cplx z2, cplx *difference)
{

  difference->real = z1.real - z2.real;
  difference->imag = z1.imag - z2.imag;
}
void cplx_multiply(const cplx *pz1,const cplx *pz2,cplx *product)
{

  product->real= (pz1->real*pz2->real) - (pz1->imag*pz2->imag);
  product->imag = (pz1->real*pz2->imag) + (pz1->imag*pz2->real);
}
```