HW3 EE599 - Computing and Software for Systems Engineers

- Unless specified: for each question that you write code
    - Provide GTest.
    - Provide runtime analysis.
    - Proof of correctness is not necessary unless specified.
- For submission, please create a zip file of all of your assignments and only submit one file.
    - PLEASE REMOVE ALL FOLDERS STARTING WITH **bazel-*** before submitting.
- Leave any extra instructions for the graders in a README file.
- Our grader should be able to call blaze run/test ... and run your code/test.
- Deadline: Monday, Feb 10th, before 6pm.
- Total: 150 points. 120 points is considered full credit.

Question 1 (10 Points. Easy)

Please compare pros and cons of the following options:

- Passing parameters by value
- Passing parameters using pointers
- Passing parameters using references
- Passing parameters using const references

Please mention when each item is preferred.

Question 2 (20 Points. Easy)

Given a vector of integers ***v***, and a number ***sum*****,** return a vector of two items which are the indices of the two numbers in ***v*** such that they add up to sum.

- If there is no answer, the return vector should be empty.
- If there are multiple answers, return any of them.

Example 1:

```
Input v = [3, 7, 11, 15], sum = 10
output: [0, 1], sum=10 (because v[0] + v[1] = 10)
```

Example 2:

```
Input v = [3, 7, 11, 15], sum = 180,
output: []
```

Example 3:

```
Input v = [1, 4, 3, 2], sum = 5,
output: either [0, 1]  OR [2, 3] is the correct answer.
```

👉**Hint:** Create a map that maps each number in v to its index.

Question 3 (60 Points. Medium)

Implement the following class for a Linked List of integer values.
All functions except for print() require a GTest.

```
struct ListNode {
 int val;
 ListNode *next;
 ListNode(int x) : val(x), next(nullptr) {}
};


class SinglyLinkedList {
Public:
  // default constructor
  SinglyLinkedList();


 // Creates a linked list out of vector "inputs" and connects the last
```

```
item's next to i(th) item in the list.
- If i is -1, the last item's next is nullptr.
- If i is greater than input size, the last item's next is nullptr.
  SinglyLinkedList(const std::vector<int> &inputs, int i)

  ~SinglyLinkedList() { } // destructor, cleans up
   bool empty(); // checks if empty
   int size(); // returns size
   void push_back(int i); // inserts at the back
   void push_front(int i); // inserts at the front
   void insert_after(ListNode* p, int i);  // inserts value i after p

   void erase(ListNode* p);  // Erases node p

   void pop_back(); // removes the first item
   void pop_back(); // removes the last item
   int back();  // returns the value of last item
   int front();  // returns the value of first item
   ListNode *GetBackPointer(); // Returns pointer to last item

   // Returns pointer to i(th) element
   ListNode *GetIthPointer(int i);

  // Prints the list: ex. Empty list: {}. List with items: {1, 2, 3}
   void print();
   ListNode *head_; // Pointer to the first element
};
```

Question 4 (20 Points. Medium)

Given an expression string, find if the input has valid brackets (i.e. { } or [ ] or ( )).

An input expression is valid if:

- Open brackets are closed by the same type of brackets.
- Open brackets must be closed in the correct order.
- An empty string is also considered valid.

You should **only** check for the validity of brackets based on the above rules, i.e. '(', ')', '[', ']', '{', '}', not the rest of the expression.

👉**Hint**: Iterate the input from beginning to end and use a **std:stack**.

```
Example 1:
Input: "(a+b)"
Output: true

Example 2:
Input: "(a+b)[c*d]{5g+h}"
Output: true

Example 3:
Input: "(a+b]"
Output: false

Example 4:
Input: "(7h+[5c)+7]"
Output: false

Example 5:
Input: "{2k+[5j]}"
Output: true

Example 6:
Input: "{2k++[5--*j]}"
Output: true
```

Question 5 (20 Points. Medium)

Write a class that stores a student's academic record. The academic record should hold marks for the following subjects:

1. Maths.
2. Computers.
3. Physics.

Requirements:

- Implement the default constructor that initializes grades to 0.
- Implement a constructor that takes the initial grades as three parameters.
- Implement the copy constructor.
- The class should support "++" and "--" operators (both postfix and prefix)
    - A "++" call should increase **all** marks of each subject by **10**.
    - A "--" call should decrease **all** marks of each subject by **20**.
- The class should support "+=" and "-=", which affect all grades of the object.
- The class should support "==" for comparison.
- After any operation, marks for any subject should stay within the range of **0** and **100**:
    - If after any operation, marks for any subject are exceeding 100 then your code should simply set the marks of that particular subject as 100.
    - Similarly, if after any operation the mark of any subject below is 0 then your code should just set the marks of that particular subject to 0.
- Print(), which returns a string that contains all marks and can later be used to print the marks.

GTest:

Create an Object and write test cases for the following scenarios:

- Check if the marks for all the subjects of the object do not go above 100 when performing increment("++") operation.
- Check if the marks for all the subjects of the object do not go below 0 when performing decrement("--") operation.
- For each operation mentioned in the question (--, ++, +-, -=) check if the marks of each subject of the object are as expected after the operations are performed. You can check the example for reference.
- Create an object of the class, say **obj1**. Further, create another object, say **obj2** and copy the contents of **obj1** to **obj2** and check if the marks of all subjects in **obj2** are similar to respective subjects in **obj1.**

Eg:

```
AcademicRecord obj1, obj2;
obj1.maths = 5;
obj1.science = 10;
obj1.physics = 95;
cout<< "Value before incrementation ::"<< obj1.print() << endl;
obj1++;
cout<< "Value after incrementation ::"<< obj1.print() << endl;
obj1--;
cout<< "Value after decrementation ::"<< obj1.print() << endl;
obj2==obj1;
cout<< "Value of Object 2::" << obj2.print() << endl;
obj2+=50;
cout<< "Value of Object 2 after increasing marks by 50 for each
subject::" << obj2.print() << endl;

obj2-=30;
cout<< "Value of Object 2 after decreasing marks by 30 for each
subject::" << obj2.print() << endl;

Output:
Value before incrementation::
Maths::5
Science::10
Physics::95
Value after incrementation::
Maths::15
Science::20
Physics::100
Value after decrementation::
Maths::0
Science::0
Physics::80
Value of Object 2::
```

```
Maths::0
Science::0
Physics::80
Value of Object 2 after increasing marks by 50 for each subject::
Maths::50
Science::50
Physics::100
Value of Object 2 after decreasing marks by 30 for each subject::
Maths::20
Science::20
Physics::70
```

Question 6 (20 Points. Medium)

Write a program that takes a vector as a parameter, prints it, and then depending upon the user input, it performs various operations on a vector using an iterator and iterator functions.

- Your code should have a variable to track the **current location** which will be pointing at the first element of the vector as soon as you start execution of your code and changes as the program runs.
- You should print a menu to the user to perform the following operations:

Example input vector: [**10, 20, 30, 40, 50, 60, 70, 80, 90, 100]**

Menu:

1. What is the **first** element?
    a. (Once this is selected, the **first** element should be printed and the current location should be set to the **first** element.)
2. What is the **last** element?
    a. (Once this is selected, the **last** element should be printed and the current location should be set to the **last** element.)
3. What is the **current element**?
    a. (This should print the value at the current location. See examples below.)
4. What is the i(th) element from the current location?
    a. (Once this is selected, the code should print the value at the current location.)

b. If the value of **i** is negative then you should prompt an appropriate message to the user and should prompt the menu options again**. (Eg: "Value of i cannot be negative")**
c. If the value of **i** is greater than the size of your vector then you should prompt an appropriate message to the user and should prompt the menu options again**. (Eg: "Value of i cannot be greater than the size of vector")**

5. Exit.

- Your code should do this until the user enters "**5**", which is "**Exit**". When the user selects **5** you should print "**Exit**!" and end the execution.
- **GTests are NOT required for this question.**
- **Submit your code, along with a sample text file of the output for this input vector:**
  - **[1, 4, 5, 23, 100, 12, 18, 175]**
  - **Assume the user selections from the menu are: 1, 2, 3, 1, 3, (4,2), 5**

Eg:

```
**************************************************************************
*
Vector: 10, 20, 30, 40, 50, 60, 70, 80, 90, 100
**************************************************************************
*
Please choose any of the following options:
   1. What is the first element?
   2. What is the last element?
   3. What is the current element?
   4. What is the ith element from the current location?
   5. Exit.
**************************************************************************
*
1
Output: 10

**************************************************************************
*
Vector: 10, 20, 30, 40, 50, 60, 70, 80, 90, 100
**************************************************************************
*
Please choose any of the following options:
```

1. What is the first element?
2. What is the last element?
3. What is the current element?
4. What is the ith element from the current location?
5. Exit.

```
**********************************************************************
*
4
Enter the value of i::
3
Output: 40

**********************************************************************
*
Vector: 10, 20, 30, 40, 50, 60, 70, 80, 90, 100
**********************************************************************
*
```

Please choose any of the following options:

1. What is the first element?
2. What is the last element?
3. What is the current element?
4. What is the ith element from the current location?
5. Exit.

```
**********************************************************************
*
3
Output: 40

**********************************************************************
*
Vector: 10, 20, 30, 40, 50, 60, 70, 80, 90, 100
**********************************************************************
*
```

Please choose any of the following options:

1. What is the first element?
2. What is the last element?
3. What is the current element?
4. What is the ith element from the current location?

5. Exit.

****
*

2

Output: 100

****
*

Vector: 10, 20, 30, 40, 50, 60, 70, 80, 90, 100

****
*

Please choose any of the following options:

1. What is the first element?

2. What is the last element?

3. What is the current element?

4. What is the ith element from the current location?

5. Exit.

****
*

4

Enter the value of i::

3

Output: Sorry! You cannot traverse 3 elements from your current location.

****
*

Vector: 10, 20, 30, 40, 50, 60, 70, 80, 90, 100

****
*

Please choose any of the following options:

1. What is the first element?

2. What is the last element?

3. What is the current element?

4. What is the ith element from the current location?

5. Exit.

****
*

```
3
Output: 100

**********************************************************************
*
Vector: 10, 20, 30, 40, 50, 60, 70, 80, 90, 100
**********************************************************************
*
Please choose any of the following options:
   1. What is the first element?
   2. What is the last element?
   3. What is the current element?
   4. What is the ith element from the current location?
   5. Exit.
**********************************************************************
*
1
Output: 10

**********************************************************************
*
Vector: 10, 20, 30, 40, 50, 60, 70, 80, 90, 100
**********************************************************************
*
Please choose any of the following options:
   1. What is the first element?
   2. What is the last element?
   3. What is the current element?
   4. What is the ith element from the current location?
   5. Exit.
**********************************************************************
*
3
Output: 10

**********************************************************************
*
Vector: 10, 20, 30, 40, 50, 60, 70, 80, 90, 100
```

```
**********************************************************************
*
Please choose any of the following options:
   1. What is the first element?
   2. What is the last element?
   3. What is the current element?
   4. What is the ith element from the current location?
   5. Exit.
**********************************************************************
*
4
Enter the value of i::
3
Output: 40


**********************************************************************
*
Vector: 10, 20, 30, 40, 50, 60, 70, 80, 90, 100
**********************************************************************
*
Please choose any of the following options:
   1. What is the first element?
   2. What is the last element?
   3. What is the current element?
   4. What is the ith element from the current location?
   5. Exit.
**********************************************************************
*
5
Exit !
```

Your code execution ends here.

Optional Questions

The goal of this section is to introduce you to more challenging questions and some common problems in coding and algorithms.

- These questions don't have any credits.
- We may not provide complete solutions or grading for them.
- Solving them is completely optional.

Optional Question 1 (Easy)

1. Write a function that prints all items in a **std::stack**. After the print, the items in the stack should remain the same.
2. Write a function that prints all items in a **std::queue**. After the print, the items in the queue should remain the same.

Optional Question 2 (Medium)

For the **SinglyLinkedList** class that you designed earlier, Implement this function:

```
bool DetectCycle();
```

This returns a true if it detects the linked list has a cycle. The cycle happens when the last item's is pointing to a node inside the list rather than **nullptr**.

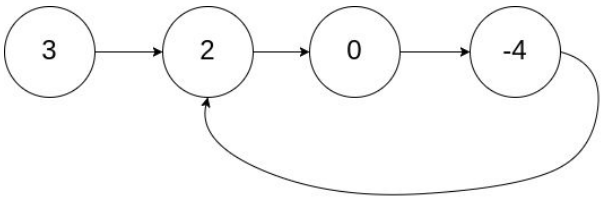
To create linked lists with cycles, use this constructor with i > 0 as described in **SinglyLinkedList** class above:

```
SinglyLinkedList(const std::vector<int> &inputs, int i)
```

Example 1:

```
Input: inputs = [3,2,0,-4], i = 1
Output: true
```

Explanation: There is a cycle in the linked list, where tail connects to the second node.

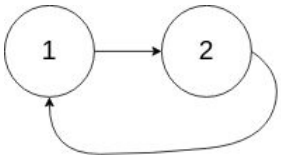


Example 2:

Input: inputs = [1,2], i = 0

Output: true

Explanation: There is a cycle in the linked list, where tail connects to the first node.



Example 3:

Input: inputs = [1], i = -1

Output: false

Explanation: There is no cycle in the linked list.

Optional Question 3 (Medium)

For the **SinglyLinkedList** class, Implement the following function:

```
void erase(ListNode* p);
```

Where p is a pointer to a node that we want to erase. **Your implementation should be O(1)**.

Optional Question 4 (Medium)

Add two more variables to the **SinglyLinkedList** class:

- size_: which tracks the size of the list.
- tail_: which always points to the last item in the list if the list doesn't have a cycle, otherwise, its value is nullptr.
- How do each of these change the runtime of the class methods?

Optional Question 5 (Medium)

Convert the **SinglyLinkedList** class to **DoublyLinkedList** class, where each node points to both its **next** and **previous** nodes. How would this change the runtime of the class methods?