The Programming Assignment Report Instructions
CSCE 221

1. The description of an assignment problem

The goal of this programming assignment was to create a vector class that used arrays, pointers and other funtions to allow for the insertion, deletion and replace capabilities that come with the STL vector class to create a dynamic "array" for characters and integers.

1. The description of data structures and algorithms used to solve the problem.

   - Provide definitions of data structures by using Abstract Data Types (ADTs)

```
class My_vec{
//member variables
int size,
capacity;
char* ptr;
public:
//member functions
My_vec();//creates a new My_vec class
~My_vec();//destructs the my vec class
My_vec(const My_vec& vec); //allows for a vector of vectors to be created
My_vec& operator=(const My_vec& vec); //allows for the copying of two vectors
int get_size() const;//returns number of elements currently within the vector
int get_capacity() const; //returns current capacity of vector (preportional to how much memory is allocated)
char& operator[](int i) const; //returns variable at element location in vector
char& operator[](int i);//returns variable at element location in vector
bool is_empty() const; //checks to see if vector is empty
char& elem_at_rank(int r) const; //returns element at specific location in vector
void insert_at_rank(int r, const char& elem); //inserts element at specific location in vector
void replace_at_rank(int r, const char& elem); //replaces element at specific location in vector
void remove_at_rank(int r); //removes element at specific location in vector
};
ostream& operator<<(ostream& out, const My_vec& vec); //displays all elements of vector
int find_max_index(const My_vec& v,int size); //finds the index of the biggest element in vector and returns it
void sort_max(My_vec& vec); //sorts the vector from biggest to smallest elements or smallest to biggest for generic
```

   For the generic vector all the instances of "char" are replaced with template "T".

   - Write about the ADTs implementation in C++class

```
class My_vec{
//member variables
int size,
capacity;
char* ptr;
public:
//member functions
My_vec();//initializes the size, capacity and vector to be used in the My_vec class
~My_vec();//dealloates memory that is not being used depending upon the size of elements in the vector
My_vec(const My_vec& vec); //accepts a reference to a vector to be placed in a slot in another vector
My_vec& operator=(const My_vec& vec); //overloads the assignment operator to accept a reference to a second vector and sets all of the elements in the second
      vector equal to the elements in the first vector using a for loop
int get_size() const;//accepts a reference to a vector and returns the number of elements currently in the vector (what size is equal to)
int get_capacity() const; //accepts a reference to a vector and returns the number of spaces available in the vector (what capacity is equal to)
char& operator[](int i) const; //overloads the [] operator to return a pointer to the reference of an element in a specific space in a vector
char& operator[](int i);//overloads the [] operator to return a pointer to the reference of an element in a specific space in a vector
bool is_empty() const; //returns a boolean value regarding whether the size of the vector is zero or not
char& elem_at_rank(int r) const; //accepts a reference to a vector as well as an integer to return a pointer to the spot in the referenced vector that the
      element refers to
void insert_at_rank(int r, const char& elem); //accepts a reference to a vector, a reference to a character and an integer to change the element in the
      location referenced by the integer to the character that was placed in the function. The size is increased by one and the elements in the spaces
      above the inserted character are moved up by one spot
void replace_at_rank(int r, const char& elem); //accepts a reference to a vector, a reference to a character and an integer to change the element in the location
      referenced by the integer to the character that was placed in the function
void remove_at_rank(int r); //accepts a reference to a vector and an integer to remove the character located in the position referred to by the integer and
      decreases the total size by one
};
ostream& operator<<(ostream& out, const My_vec& vec); //overloads the << operator and uses a reference to a vector and a for loop to display all the elements in all
      slots of the vector containing an element
int find_max_index(const My_vec& v,int size); //accepts a reference to a vector and an integer for the size of the vector to use a for loop to go through the size
```

```
    of the vector. The location for the character with the highest ascii value is stored and returned
void sort_max(My_vec& vec); //accepts a reference to a vector and uses the replace_at_rank and find_max_index functions to swap the elements of the vector
    until the vector is sorted from highest to lowest, or lowest to highest for generic.
```

For the generic vector all the instances of “char” are replaced with template “T”.

- Describe algorithms used to solve the problem.

There are three main algorithms used in this program to accomplish the problem

```
for (int i = size - 1; i >= r ;i--)
ptr[i+1]=ptr[i];
ptr[r] = elem;
if (ptr[r] != '\0')
size++;
```

The above algorithm is used in the insert_at_rank function to move up all the elements with a higher index than the element being inserted.

```
for (int i = v.get_size()- size;
i <= v.get_capacity(); i++)
{
if (v[i] > letter)
{
letter = int(v[i]);
location = i;
}
}
```

The above algorithm is used in the find_max_index function to sort through the given vector and hold the location of the highest letter or number it passes in the for loop.

```
for (int i = 0; i <= vec.get_size() - 1; i++){
int max = find_max_index(vec, vec.get_size()-i);
char k = vec[max];
vec.replace_at_rank(max,vec[i]);
vec.replace_at_rank(i,k);
}
```

The above algorithm is used in the sort_max_index function to swap elements with the highest index until the final vector is sorted from highest to lowest.

- Analyze the algorithms according to assignment requirements.

The first algorithm will be used anytime that an element is being inserted in a space lower than the size of the vector. If it is the first element being added to a vector or being added to the end of a vector, the funciton will not have to iterate the for loop. The second algorithm will iterate the most if the vector is ordered from lowest to highest. The if statement will return as true as many times as the size of the vector. However if the vector is already ordered from highest to lowest the if statement will only iterate once. The third algorithm follows the same format as the second algorithm, however it iterates the same amount of times with every call. In the case that the vector is already sorted from highest to lowest, the function will replace the element with itself regardless if it is the same element.

2. A C++ organization and implementation of the problem solution

   - Provide a list and description of classes or interfaces used by a program such as classes used to implement the data structures or exceptions.

   -My_vec-creates a dynamic vector of character objects using arrays and a multitude of functions to manipulate the vector.

   -My_genvec-creates a dynamic vector of objects specified by the template using arrays and a multitude of functions to manipulate the vector.

   - Include in the report the class declarations from a header file (.h) and their implementation from a source file (.cpp).

   -For the original My_vec class, the class and class functions are declared in the .h file and constructed in the .cpp file, however for the generic My_vec the class is defined and constructed in the .h file since templates eliminate the ability to use multiple files.

   - Provide features of the C++ programming paradigms like Inheritance or Polymorphism in case of object oriented programming, or Templates in the case of generic programming used in your implementation.

   In the instance of object-oriented programming the paradigms of inheritance, encapsulation and polymorphism are seen in the program of My_vec. Encapsulation is seen by the private members of the My_vec class including the size, capacity and ptr variables. This act of data hiding is important so the user does not have the capability to change these variables without using the public functions of My_vec. Furthermore, inheritance is used in the My_vec.cpp file in the find_max_index, sort_max and << operator overload functions. They inherit an instance of the My_vec class to return a seperate value in the main function. Finally, polymorphism is used in the implementation of the My_vec class by having the ability to take a pointer to two seperate vectors containing different element types and be able to change them with the same function. In generic programming, templates were very useful when greating the My_genvec class. Rather than making a different class for each variable type, only one class needed to be made. In order to change the varibable type the user simply needed to specify the type in the vector angle brackets.

3. A user guide description how to navigate your program with the instructions how to:

   - compile the program: specify the directory and file names, etc.

   The directory is Kaiser-Alexander-A1

   The file names for the original My_vec are: My_vec.h, My_vec.cpp, main.cpp and makefile; compile with command make

   The file names for the generic My_vec are: My_genvec.h genmain.cpp, and genmakefile.mak; compile with command make -f genmakefile.mak

   —run the program: specify the name of an executable file.

   To run the original My_vec use command ./main

   To run the generic My_vec use command ./genmain

4. Specifications and description of input and output formats and files

   - The type of files: keyboard, text files, etc (if applicable).

   The program requires that data is written/changed in the main function of each .cpp file in order to change the output of the program. The output of the program comes from running the main file after compiling in g++ in an SSH client.

   - A file input format: when a program requires a sequence of input items, specify the number of items per line or a line termination. Provide a sample of a required input format.

   The program requires for user input to transform the main function before compliling. Thus, input should be made within the functions in the main function. Each function should take up one line increase readability of the code.

   - Discuss possible cases when your program could crash because of incorrect input (a wrong file name, strings instead of a number, or such cases when the program expects 10 items to read and it finds only 9.)

   The program knows to throw an exception when attempting to insert or remove an item with an index greater than the size of the vector. Furthermore, if the user attempts to insert a variable that is different than the type specified in the template the program will simply refuse to complile.

5. Provide types of exceptions and their purpose in your program.

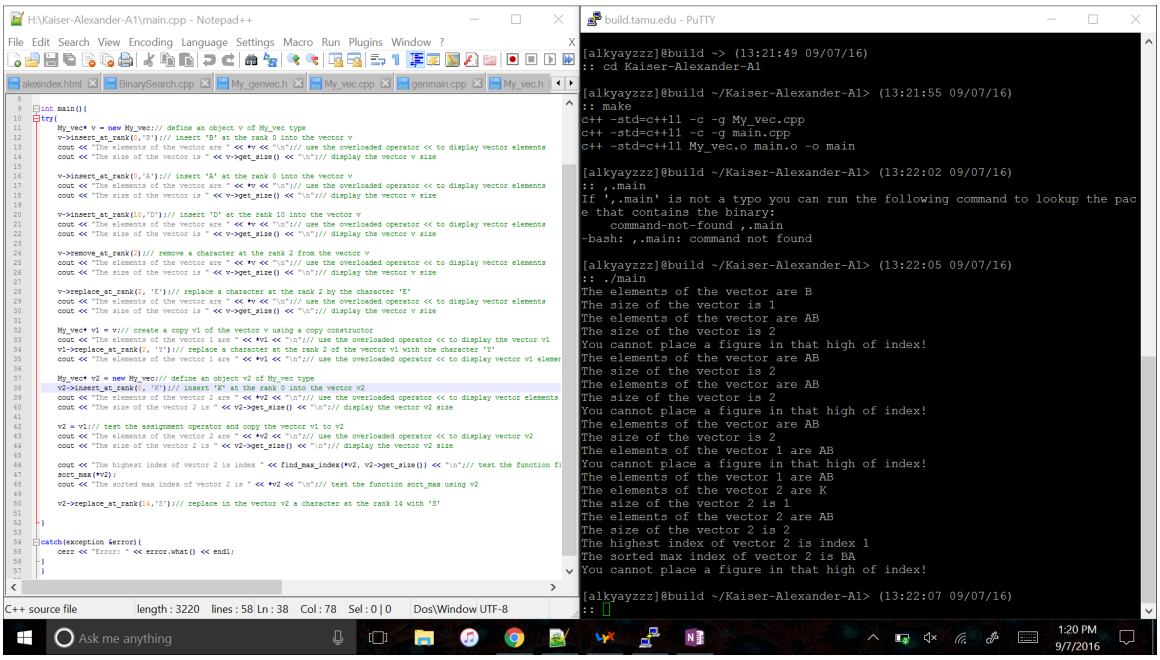
   - logical exceptions (such as deletion of an item from an empty container, etc.).

   In order to prevent against logical exceptions, a series of if statements are implemented within the code to terminate the program in the event that it occurs. An example includes the insert_at_rank function. The if statement that is placed at the beginning of the code terminates the program when attempting to insert an element into a space that contains an empty character. A second example includes the remove_at_rank function. If the user attempts to remove an element that contains an empty character (doesn't exist) then the size of the vector is not changed.

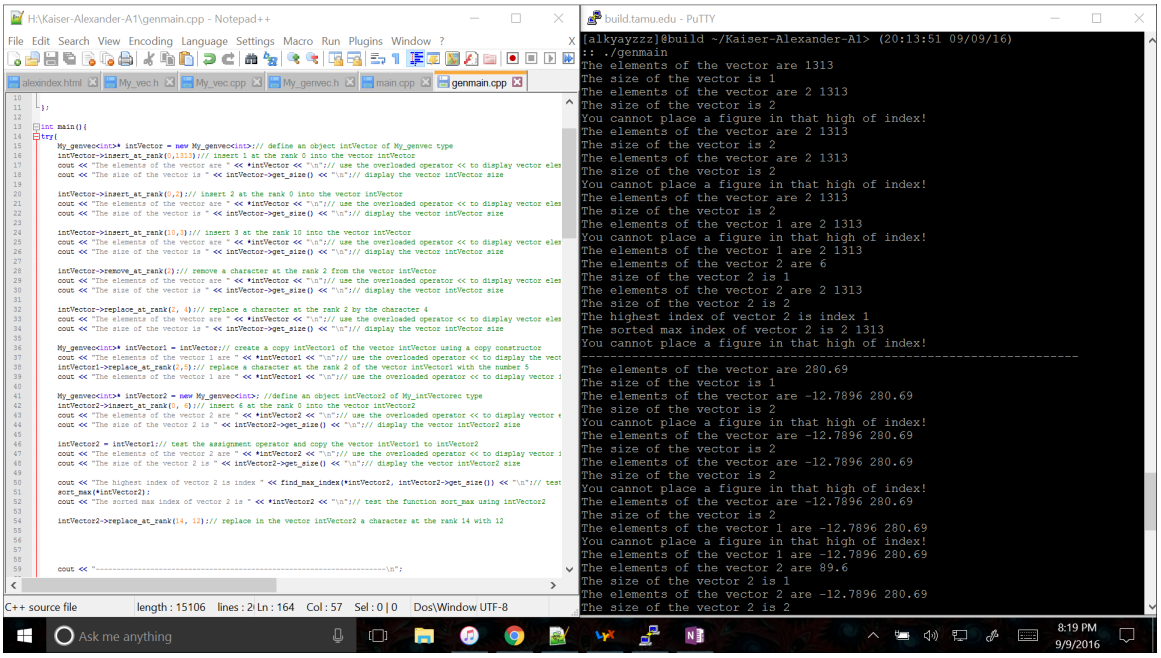
   - runtime exception (such as division by 0, etc.)

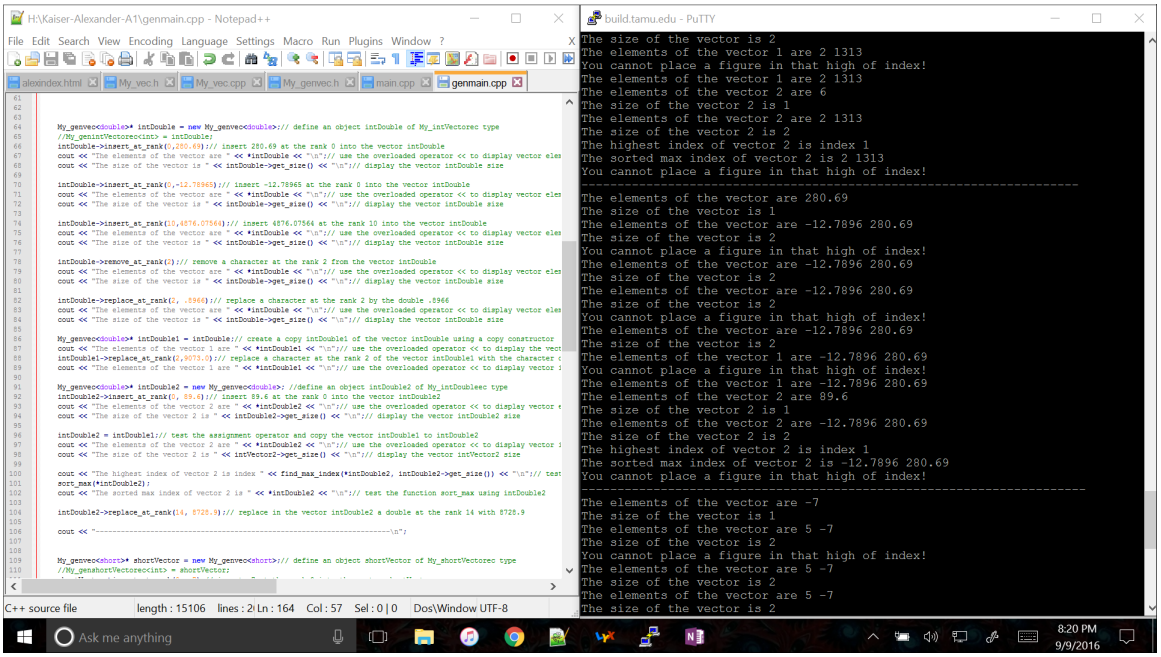
   The prevention of runtime exceptions comes from the try-catch statements in the main function. When a runtime exception is encountered (such as a segmentation fault) the program is terminated completely and a reason is given as to why the termination occurred.

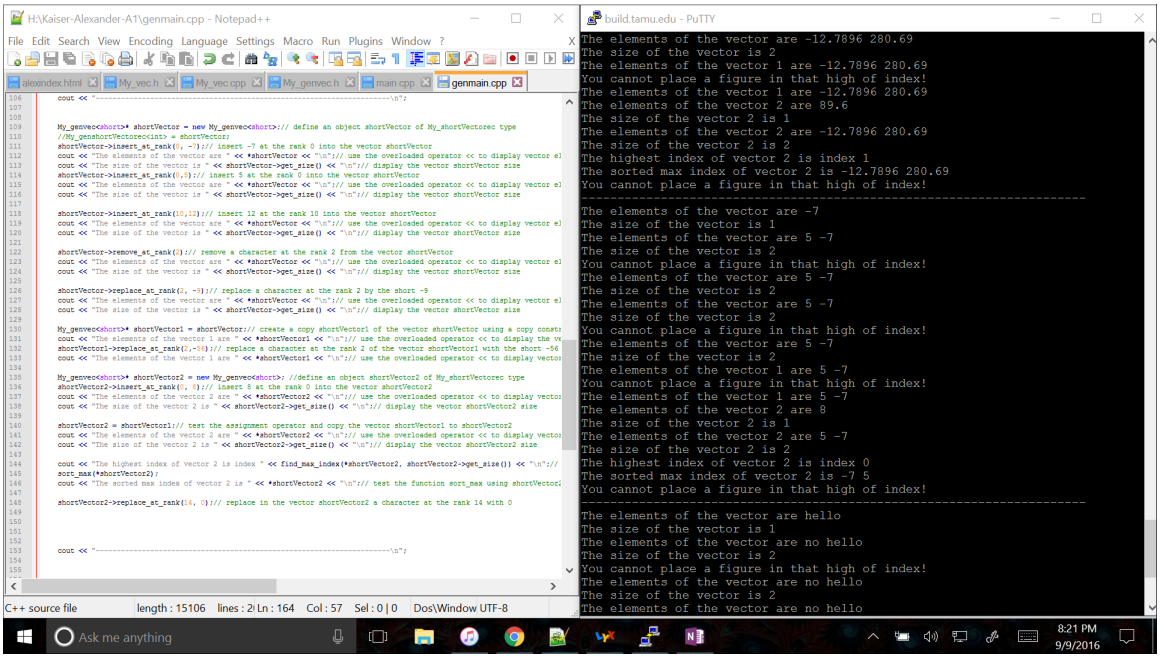
6. Test your program for correctness using valid, invalid, and random inputs (e.g., insertion of an item at the beginning, at the end, or at a random place into a sorted vector). Include evidence of your testing, such as an output file or screen shots with an input and the corresponding output.

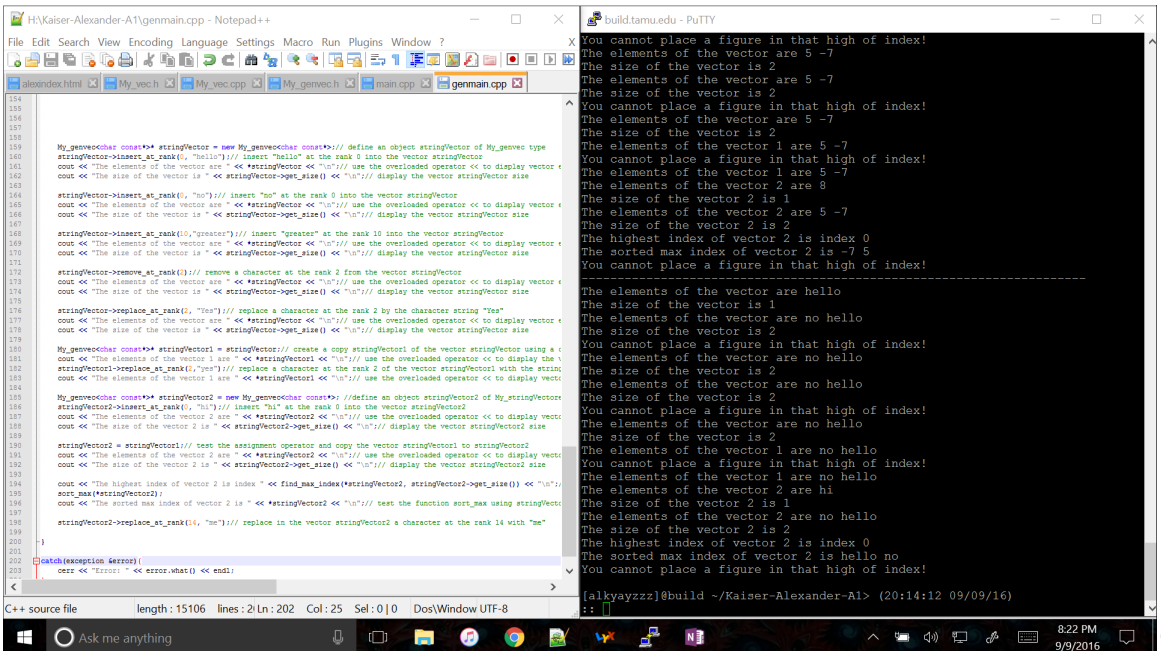


The original My_vec class is displayed above. When attempting to place an element in a location that does not yet exist an error is thrown and the user is prompted with the error. The instructions given in the main were followed showing the input of correct, incorrect and random values.




H:\Kaiser-Alexander-A1\genmain.cpp - Notepad++
build.tamu.edu - PuTTY
The size of the vector is 2
The elements of the vector 1 are 2 1313
You cannot place a figure in that high of index!
The elements of the vector 1 are 2 1313
The elements of the vector 2 are 6
The size of the vector 2 is 1
The elements of the vector 2 are 2 1313
The size of the vector 2 is 2
The highest index of vector 2 is index 1
The sorted max index of vector 2 is 2 1313
You cannot place a figure in that high of index!
The elements of the vector are 280.69
The size of the vector is 1
The elements of the vector are -12.7896 280.69
The size of the vector is 2
You cannot place a figure in that high of index!
The elements of the vector are -12.7896 280.69
The size of the vector is 2
The elements of the vector are -12.7896 280.69
The size of the vector is 2
You cannot place a figure in that high of index!
The elements of the vector are -12.7896 280.69
The size of the vector is 2
The elements of the vector 1 are -12.7896 280.69
You cannot place a figure in that high of index!
The elements of the vector 1 are -12.7896 280.69
The elements of the vector 2 are 89.6
The size of the vector 2 is 1
The elements of the vector 2 are -12.7896 280.69
The size of the vector 2 is 2
The highest index of vector 2 is index 1
The sorted max index of vector 2 is -12.7896 280.69
You cannot place a figure in that high of index!
The elements of the vector are -7
The size of the vector is 1
The elements of the vector are 5 -7
The size of the vector is 2
You cannot place a figure in that high of index!
The elements of the vector are 5 -7
The size of the vector is 2
The elements of the vector are 5 -7
The size of the vector is 2
8:20 PM
9/9/2016


H:\Kaiser-Alexander-A1\genmain.cpp - Notepad++
build.tamu.edu - PuTTY
The elements of the vector are -12.7896 280.69
The size of the vector is 2
The elements of the vector 1 are -12.7896 280.69
You cannot place a figure in that high of index!
The elements of the vector 1 are -12.7896 280.69
The elements of the vector 2 are 89.6
The size of the vector 2 is 1
The elements of the vector 2 are -12.7896 280.69
The size of the vector 2 is 2
The highest index of vector 2 is index 1
The sorted max index of vector 2 is -12.7896 280.69
You cannot place a figure in that high of index!
The elements of the vector are -7
The size of the vector is 1
The elements of the vector are 5 -7
The size of the vector is 2
You cannot place a figure in that high of index!
The elements of the vector are 5 -7
The size of the vector is 2
The elements of the vector are 5 -7
The size of the vector is 2
You cannot place a figure in that high of index!
The elements of the vector are 5 -7
The elements of the vector 1 are 5 -7
You cannot place a figure in that high of index!
The elements of the vector 1 are 5 -7
The elements of the vector 2 are 8
The size of the vector 2 is 1
The elements of the vector 2 are 5 -7
The size of the vector 2 is 2
The highest index of vector 2 is index 0
The sorted max index of vector 2 is -7 5
You cannot place a figure in that high of index!
The elements of the vector are hello
The size of the vector is 1
The elements of the vector are no hello
The size of the vector is 2
You cannot place a figure in that high of index!
The elements of the vector are no hello
The size of the vector is 2
The elements of the vector are no hello
8:21 PM
9/9/2016



The three screen clippings taken above show the My_genvec class at work. The same instructions given in the main.cpp were followed and exceptions were thrown if there was an incorrect input.