

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 functions 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 (proportional 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(); //deallocates 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 function 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 separate 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 separate vectors containing different element types and be able to change them with the same function. In generic programming, templates were very useful when creating 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 variable 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 compiling. 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 compile.

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.


```
H:\Kaiser-Alexander-A1\genmain.cpp - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
alexindex.html My_vec.h My_vec.cpp My_genvec.h main.cpp genmain.cpp
81 //My_genvec::intDouble = intDouble;
82 //My_genvec::intDouble = intDouble;
83
84 My_genvec::intDouble = new My_genvec::intDouble; // define an object intDouble of My_intVecBase type
85 //My_genvec::intDouble = intDouble;
86 intDouble->insert_at_rank(0,200.69); // insert 200.69 at the rank 0 into the vector intDouble
87 cout << "The elements of the vector are " << *intDouble << "\n"; // use the overloaded operator << to display vector else
88 cout << "The size of the vector is " << intDouble->get_size() << "\n"; // display the vector intDouble size
89
90 intDouble->insert_at_rank(0,-12.7896); // insert -12.7896 at the rank 0 into the vector intDouble
91 cout << "The elements of the vector are " << *intDouble << "\n"; // use the overloaded operator << to display vector else
92 cout << "The size of the vector is " << intDouble->get_size() << "\n"; // display the vector intDouble size
93
94 intDouble->insert_at_rank(10,476.07544); // insert 476.07544 at the rank 10 into the vector intDouble
95 cout << "The elements of the vector are " << *intDouble << "\n"; // use the overloaded operator << to display vector else
96 cout << "The size of the vector is " << intDouble->get_size() << "\n"; // display the vector intDouble size
97
98 intDouble->remove_at_rank(1); // remove a character at the rank 2 from the vector intDouble
99 cout << "The elements of the vector are " << *intDouble << "\n"; // use the overloaded operator << to display vector else
100 cout << "The size of the vector is " << intDouble->get_size() << "\n"; // display the vector intDouble size
101
102 intDouble->replace_at_rank(1,5946); // replace a character at the rank 2 by the double 5946
103 cout << "The elements of the vector are " << *intDouble << "\n"; // use the overloaded operator << to display vector else
104 cout << "The size of the vector is " << intDouble->get_size() << "\n"; // display the vector intDouble size
105
106 My_genvec::intDouble1 = intDouble; // create a copy intDouble1 of the vector intDouble using a copy constructor
107 cout << "The elements of the vector 1 are " << *intDouble1 << "\n"; // use the overloaded operator << to display the vect
108 intDouble1->replace_at_rank(0,907.9); // replace a character at the rank 0 of the vector intDouble1 with the character 9
109 cout << "The elements of the vector 1 are " << *intDouble1 << "\n"; // use the overloaded operator << to display vector 1
110
111 My_genvec::intDouble2 = new My_genvec::intDouble; //define an object intDouble2 of My_intVecBase type
112 intDouble2->insert_at_rank(0,89.6); // insert 89.6 at the rank 0 into the vector intDouble2
113 cout << "The elements of the vector 2 are " << *intDouble2 << "\n"; // use the overloaded operator << to display vector 2
114 cout << "The size of the vector 2 is " << intDouble2->get_size() << "\n"; // display the vector intDouble2 size
115
116 intDouble2 = intDouble; // test the assignment operator and copy the vector intDouble2 to intDouble2
117 cout << "The elements of the vector 2 are " << *intDouble2 << "\n"; // use the overloaded operator << to display vector 2
118 cout << "The size of the vector 2 is " << intDouble2->get_size() << "\n"; // display the vector intDouble2 size
119
120 cout << "The highest index of vector 2 is index " << find_max_index(*intDouble2, intDouble2->get_size()) << "\n"; // test
121 sort_max(intDouble2);
122 cout << "The sorted max index of vector 2 is " << *intDouble2 << "\n"; // test the function sort_max using intDouble2
123
124 intDouble2->replace_at_rank(14,8725.9); // replace in the vector intDouble2 a double at the rank 14 with 8725.9
125
126 cout << "-----\n";
127
128 My_genvec::shortVector = new My_genvec::shortVector; // define an object shortVector of My_shortVecBase type
129 //My_genvec::shortVector = shortVector;
130 shortVector->insert_at_rank(0,-7); // insert -7 at the rank 0 into the vector shortVector
131 cout << "The elements of the vector are " << *shortVector << "\n"; // use the overloaded operator << to display vector el
132 cout << "The size of the vector is " << shortVector->get_size() << "\n"; // display the vector shortVector size
133 shortVector->insert_at_rank(0,89.6); // insert 89.6 at the rank 0 into the vector shortVector
134 cout << "The elements of the vector are " << *shortVector << "\n"; // use the overloaded operator << to display vector el
135 cout << "The size of the vector is " << shortVector->get_size() << "\n"; // display the vector shortVector size
136
137 shortVector->insert_at_rank(10,23); // insert 23 at the rank 10 into the vector shortVector
138 cout << "The elements of the vector are " << *shortVector << "\n"; // use the overloaded operator << to display vector el
139 cout << "The size of the vector is " << shortVector->get_size() << "\n"; // display the vector shortVector size
140
141 shortVector->remove_at_rank(1); // remove a character at the rank 2 from the vector shortVector
142 cout << "The elements of the vector are " << *shortVector << "\n"; // use the overloaded operator << to display vector el
143 cout << "The size of the vector is " << shortVector->get_size() << "\n"; // display the vector shortVector size
144
145 shortVector->replace_at_rank(1,-9); // replace a character at the rank 2 by the short -9
146 cout << "The elements of the vector are " << *shortVector << "\n"; // use the overloaded operator << to display vector el
147 cout << "The size of the vector is " << shortVector->get_size() << "\n"; // display the vector shortVector size
148
149 My_genvec::shortVector1 = shortVector; // create a copy shortVector1 of the vector shortVector using a copy const
150 cout << "The elements of the vector 1 are " << *shortVector1 << "\n"; // use the overloaded operator << to display the v
151 shortVector1->replace_at_rank(0,-54); // replace a character at the rank 0 of the vector shortVector1 with the short -54
152 cout << "The elements of the vector 1 are " << *shortVector1 << "\n"; // use the overloaded operator << to display vector
153
154 My_genvec::shortVector2 = new My_genvec::shortVector; //define an object shortVector2 of My_shortVecBase type
155 shortVector2->insert_at_rank(0,8); // insert 8 at the rank 0 into the vector shortVector2
156 cout << "The elements of the vector 2 are " << *shortVector2 << "\n"; // use the overloaded operator << to display vector
157 cout << "The size of the vector 2 is " << shortVector2->get_size() << "\n"; // display the vector shortVector2 size
158
159 shortVector2 = shortVector1; // test the assignment operator and copy the vector shortVector2 to shortVector2
160 cout << "The elements of the vector 2 are " << *shortVector2 << "\n"; // use the overloaded operator << to display vector
161 cout << "The size of the vector 2 is " << shortVector2->get_size() << "\n"; // display the vector shortVector2 size
162
163 cout << "The highest index of vector 2 is index " << find_max_index(*shortVector2, shortVector2->get_size()) << "\n"; // test
164 sort_max(shortVector2);
165 cout << "The sorted max index of vector 2 is " << *shortVector2 << "\n"; // test the function sort_max using shortVector2
166
167 shortVector2->replace_at_rank(14,0); // replace in the vector shortVector2 a character at the rank 14 with 0
168
169 cout << "-----\n";
170
171 C++ source file length: 15106 lines : 2 Ln: 164 Col: 57 Sel: 0 | 0 Dos:Window UTF-8
```

```
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

```
My_genvec<char> const* stringVector = new My_genvec<char> const*;// define an object stringVector of My_genvec type
stringVector->insert_at_rank(0, "hello");// insert "hello" at the rank 0 into the vector stringVector
cout << "The elements of the vector are " << *stringVector << "\n";// use the overloaded operator << to display vector
cout << "The size of the vector is " << stringVector->get_size() << "\n";// display the vector stringVector size

stringVector->insert_at_rank(0, "no");// insert "no" at the rank 0 into the vector stringVector
cout << "The elements of the vector are " << *stringVector << "\n";// use the overloaded operator << to display vector
cout << "The size of the vector is " << stringVector->get_size() << "\n";// display the vector stringVector size

stringVector->insert_at_rank(0, "greater");// insert "greater" at the rank 0 into the vector stringVector
cout << "The elements of the vector are " << *stringVector << "\n";// use the overloaded operator << to display vector
cout << "The size of the vector is " << stringVector->get_size() << "\n";// display the vector stringVector size

stringVector->remove_at_rank(1)// remove a character at the rank 2 from the vector stringVector
cout << "The elements of the vector are " << *stringVector << "\n";// use the overloaded operator << to display vector
cout << "The size of the vector is " << stringVector->get_size() << "\n";// display the vector stringVector size

stringVector->replace_at_rank(2, "tea");// replace a character at the rank 2 by the character string "tea"
cout << "The elements of the vector are " << *stringVector << "\n";// use the overloaded operator << to display vector
cout << "The size of the vector is " << stringVector->get_size() << "\n";// display the vector stringVector size

My_genvec<char> const* stringVector1 = new My_genvec<char> const*;// define an object stringVector1 of My_genvec type
stringVector1->insert_at_rank(0, "hi");// insert "hi" at the rank 0 into the vector stringVector1
cout << "The elements of the vector 1 are " << *stringVector1 << "\n";// use the overloaded operator << to display vector
cout << "The size of the vector 1 is " << stringVector1->get_size() << "\n";// use the overloaded operator << to display vector

stringVector1->replace_at_rank(0, "yes");// replace a character at the rank 2 of the vector stringVector1 with the string
cout << "The elements of the vector 1 are " << *stringVector1 << "\n";// use the overloaded operator << to display vector

My_genvec<char> const* stringVector2 = new My_genvec<char> const*;// define an object stringVector2 of My_genvec type
stringVector2->insert_at_rank(0, "hi");// insert "hi" at the rank 0 into the vector stringVector2
cout << "The elements of the vector 2 are " << *stringVector2 << "\n";// use the overloaded operator << to display vector
cout << "The size of the vector 2 is " << stringVector2->get_size() << "\n";// use the overloaded operator << to display vector

stringVector2 = stringVector1// test the assignment operator and copy the vector stringVector1 to stringVector2
cout << "The elements of the vector 2 are " << *stringVector2 << "\n";// use the overloaded operator << to display vector
cout << "The size of the vector 2 is " << stringVector2->get_size() << "\n";// use the overloaded operator << to display vector

cout << "The highest index of vector 2 is index " << find_max_index(*stringVector2, stringVector2->get_size()) << "\n";//
sort_max(*stringVector2);
cout << "The sorted max index of vector 2 is " << *stringVector2 << "\n";// test the function sort_max using stringVector2

stringVector2->replace_at_rank(14, "me");// replace in the vector stringVector2 a character at the rank 14 with "me"

}

catch(exception &error)
{
    cerr << "Error: " << error.what() << endl;
}
```

```
alkyayzzz@build ~/Kaiser-Alexander-A1> (20:14:12 09/09/16)
```

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.