

# Homework #2

Name: Purnabhishek Sripathi

Email: [ps747@nau.edu](mailto:ps747@nau.edu)

Userid: 6274051

## Problem #1 (of 1)

### 1. A default constructor (that zeroes everything out)

```
99 class Queries_AR
100 {
101 public:
102     vector<array<char,32>> queries; // Vector for query storage, with each query being a 32-character array
103     long long int size; //Declaring the variable to store the size
104
105     Queries_AR() // Default Constructor
106     {
107     }
108 }
109 Queries_AR(char *path) //Custom Constructor that takes file path as input and retrieves queries from a text document
110 {
111     long long int fileSize = get_size(path); // getting the size of the file
112     cout<<"queries filesize:"<<fileSize<<endl; // printing the size of the file
113     // fflush(stdout);
114     queries.reserve(fileSize/45); //Reserving the space in the vector by file size & expected entry length
115     // cout<<"check1 \n";
116     readDataSet(path);
117 }
118 }
```

### 2. At least one custom constructor (e.g. one taking a file path or ifstream as input)

```
99 class Queries_AR
100 {
101 public:
102     vector<array<char,32>> queries; // Vector for query storage, with each query being a 32-character array
103     long long int size; //Declaring the variable to store the size
104
105     Queries_AR() // Default Constructor
106     {
107     }
108 }
109 Queries_AR(char *path) //Custom Constructor that takes file path as input and retrieves queries from a text document
110 {
111     long long int fileSize = get_size(path); // getting the size of the file
112     cout<<"queries filesize:"<<fileSize<<endl; // printing the size of the file
113     // fflush(stdout);
114     queries.reserve(fileSize/45); //Reserving the space in the vector by file size & expected entry length
115     // cout<<"check1 \n";
116     readDataSet(path);
117 }
118 }
```

### 3. A function to read the query dataset file.

```
108     }
109     Queries_AR(char *path) //Custom Constructor that takes file path as input and retrieves queries from a text document
110     {
111         long long int fileSize = get_size(path); // getting the size of the file
112         cout<<"queries fileSize:"<<fileSize<<endl; // printing the size of the file
113         // fflush(stdout);
114         queries.reserve(fileSize/45); //Reserving the space in the vector by file size & expected entry length
115         // cout<<"check1 \n";
116         readDataSet(path);
117     }
118     void readDataSet(char* path){ // Function to read the query dataset file
119     {
120         ifstream queryFile(path); // opening the file with given path
121         int fileSize;
122         string currLine; //Declaring the string currLine to store each line read from the file
123         if (!queryFile.is_open())
124             cout << "err: unable to open the file"; //print the error if the file cannot be opened
125         bool val=false; //Declaring the boolean variable to alternate reading lines
126         // cout<<"check2 \n";
127         while (getline(queryFile, currLine)){ // while loop to read the lines from the file
128             val=!val;
129             if(val)
130                 continue;
131             array<char,32> arr; // array to store the current query
132             fileSize=32;
133             while(--fileSize){
134                 arr[32-fileSize]=currLine[32-fileSize]; //Copying the chars from currLine to array
135             }
136             queries.emplace_back(arr); //Adding query to vector of queries
137         }
138         queries.shrink_to_fit(); //Shrink the vector to fit the actual number of elements
139         // cout<<"check3 \n";
140         fflush(stdout);
141         cout << "\nqueries read successfully: " << queries.size(); // Printing the no:of queries read
142         queryFile.close(); // Closing the file
143     }
144 }
```

### 4. A search function designed to find a sequence fragment within class's data.

```
144     }
145     bool linear_search(char *c){ // Linear search for a query
146     {
147         int temp;
148         for(int i=0;i<queries.size();++i){ // Iterating through the queries
149             temp=-1;
150             while((++temp)<32&&queries[i][temp]==c[temp]); // Comparing the each character of the query
151             if(temp==32)
152                 return true; // Returning true if all characters match
153         }
154         return false; // Returning false if no match is found
155     }
156     bool binary_search(char *c){ // Binary search for a query in the vector
157     {
158         long long int start=0,end=queries.size();
159         int temp; //declaring the temp variable
160         while(start<=end){
161             long long int mid= (end-start)/2+start; // calculating the middle index
162             temp=-1; // initializing the temp to -1
163             while((++temp)<32&&queries[mid][temp]==c[temp]); // while loop to compare each character of the query
164             if(temp==32) // if statement to return true if all characters match
165                 return true;
166             else if(queries[mid][temp]<c[temp]) // Move to the right half
167                 start=mid+1;
168             else // Move to the left half
169                 end=mid-1;
170         }
171         return false; //return false if no match is found
172     }
173 }
```

## 5. A function to sort the fragments of the Queries\_AR object.

```
170     }
171     void sortQueries(){ // function to sort the fragments of the Queries_AR object.
172         // cout<<"before sort"<<endl;
173         // for(int i=0;i<10;++i){
174             // cout<<string(queries[i].begin(),queries[i].end())<<endl;
175         // }
176         //sort(queries.begin(),queries.end());
177         // cout<<"after sort"<<endl;
178         // for(int i=0;i<10;++i){
179             // cout<<string(queries[i].begin(),queries[i].end())<<endl;
180         // }
181         merge_sort(0, queries.size() - 1, queries); // merge sort method calling
182     }
183     ~Queries_AR(){} // queries_ar destructor
184 };
185
```

## 6. A destructor

```
183     ~Queries_AR(){} // queries_ar destructor }
184 };
185
186 class GNOME
187 {
188     public:
189     char *data = nullptr; // data pointer to store the gnome data
190     long long int size = 0; // size variable to store the total size of the data
191     GNOME(char *path) // Constructor that reads gnome data from a file
192     {
193         unsigned long long fileSize = get_size(path); // Getting size of the file and storing it in fileSize
194         cout<<"\ngnome size"<< fileSize<<endl; // Printing the size of the file
195         fflush(stdout); // Flush the output buffer
196         data = (char *)malloc(fileSize); // Allocating memory for the gnome data
197         ifstream queryFile(path); // Open the file with the given path
198         string currLine; // declaring string to store the each line read from the file
199         currLine.reserve(100); // reversing the space for 100 characters
200         if (!queryFile.is_open()) // condition to check if the file failed to open
201             cout << "err: unable to open the file"; // print error message
202         while (getline(queryFile, currLine)) //reading the file line by line until end of the file
203         {
204             if (currLine[0] != '>'){ // condition to check is the line doesnot start with >
205                 memcpy(data + size, currLine.c_str(), currLine.size()); //copying the current line content
206                 size+=currLine.size(); // incrementing the size
207             }
208         }
209         cout<<"total chars read from gnome file:"<<size<<endl; // printing the total chars read from the gnome file
210         queryFile.close(); // closing the file
211     }
212     ~GNOME() // destructor to free the allocated memory for gnome
213     {
214         free(data); // free the allocated memory
215     }
216 };
```

A. (30 pts)

1. How long did it take you to search for the first 10K, 100K, and 1M 32-character long fragments of the subject dataset within the query dataset?

**Ans:**

**Time taken to search for the first 10K:**

18624 seconds

Approximately it takes 5.1733333 hours.

**Time taken to search for the first 100K:**

181741 seconds

Approximately it takes 50.4836111 hours.

**Time taken to search for the first 1M:** 504.836111 hours approx.

I have calculated the approximate calculation for 1M is as below:

Time taken to search for the first 1M = (Time taken for the first 100K fragments) \* (1M / 100K)

Time taken to search for first 1M fragments = 50.4836111 hours \* (1,000,000 / 100,000)

Time taken to search for first 1M fragments = 50.4836111 hours \* 10

Time taken to search for first 1M fragments = 504.836111 (**21 days approximately**)

2. How long would it take to search for every possible 32-character long fragment of the subject dataset within the query dataset? Please note that depending on the efficiency of your algorithm, this step may take a long time. If the total time is greater than 24 CPU hours, provide an estimate rather than an exact number.

**Ans:**

The time required for every possible 32-character long fragment within the query data set, it takes approximately **1543378.21 hours**.

To translate it in days it would take **64307.43 days** and if we converted it into years, it would take **176.18 years** to compute.

As I have used Linear search here takes these many years to run the program.

3. Print the 20 query fragments found within the subject dataset that have the largest indices (i.e. found later in the subject) for the first 10K, 100K, and 1M 32-character long fragments.

Screenshot that shows the last 20 fragments for 10k:

```
ondemand.hpc.nau.edu/pun/sys/dashboard/files/fs//home/ps747/Homework2/Isds_hw2_output_A1

queries filesize:5188888898

queries read successfully: 100000000
gnome size3095411481
total chars read from gnome file:3057186663

Question: A
time taken for first 10k: 18624 sec
fragments found in first 10k
CCAGCAAACAAAGGAAATAACCAAGATCAGAG
AACAAAGGAAATAACCAAGATCAGAGCAGAAC
AAAGGAAATAACCAAGATCAGAGCAGAACTAA
AAGGAAATAACCAAGATCAGAGCAGAACTAAA
AGGAAATAACCAAGATCAGAGCAGAACTAAAT
GGAAATAACCAAGATCAGAGCAGAACTAAATG
GAAATAACCAAGATCAGAGCAGAACTAAATGA
AAATAACCAAGATCAGAGCAGAACTAAATGAA
AATAACCAAGATCAGAGCAGAACTAAATGAAA
ATAACCAAGATCAGAGCAGAACTAAATGAAAT
TAACCAAGATCAGAGCAGAACTAAATGAAATT
AACCAAGATCAGAGCAGAACTAAATGAAATTG
ACCAAGATCAGAGCAGAACTAAATGAAATTGA
AAGATCAGAGCAGAACTAAATGAAATTGACAC
AGATCAGAGCAGAACTAAATGAAATTGACACA
CTAAATGAAATTGACACAACAACAACAAC
AAACATAAATAAAACAAAAATTTGGTTATTTG
CCAAACCCCAATCCCAGCAAACAAAGGAAATAA
AACCCAATCCCAGCAAACAAAGGAAATAACCA
CAATCCCAGCAAACAAAGGAAATAACCAAGAT
```

Screenshot that shows the last 20 fragments for 10k:

```
← → ↻ 🏠 📄 ondemand.hpc.nau.edu/pun/sys/dashboard/files/fs//home/ps747/Homework2/lsds_hw2_output_A1
AAATAACCAAGATCAGAGCAGAACTAAATGAA
AATAACCAAGATCAGAGCAGAACTAAATGAAA
ATAACCAAGATCAGAGCAGAACTAAATGAAAT
TAACCAAGATCAGAGCAGAACTAAATGAAATT
AACCAAGATCAGAGCAGAACTAAATGAAATTG
ACCAAGATCAGAGCAGAACTAAATGAAATTGA
AAGATCAGAGCAGAACTAAATGAAATTGACAC
AGATCAGAGCAGAACTAAATGAAATTGACACA
CTAAATGAAATTGACACAACAACAACAAC
AAACATAAAATAAAACAAAAATTTGGTTATTTG
CCAAACCAATCCCAGCAAACAAAGGAAATAA
AACCCAATCCCAGCAAACAAAGGAAATAACCA
CAATCCCAGCAAACAAAGGAAATAACCAAGAT
time taken for first 100k: 181741 sec
fragments found in first 100k
ACTACCCAACATGGTAGCCAGTGGTCACTAAT
GCAGATACAGGACATTCCCGTTATCTCAGAGA
GAAAAAGAGGAGCTAGGTGGATGAGGAGTGTA
CTCAGTGGCCATATTTCTCAATGTGCAGCCA
TTTCTCTCTTACAGAGTTTGAAATGCTGCTTG
TGATTGAATGATGATGCCTCACTTTCACATAC
TGAATGATGATGCCTCACTTTCACATACAGAT
AGCCCAACAAGATTCCAATAGATGTGGTGTTA
AGTGCTAGGATTACAGGTGTGAGCCACTGTGC
GTGCTAGGATTACAGGTGTGAGCCACTGTGCC
TGCTAGGATTACAGGTGTGAGCCACTGTGCCC
GCTAGGATTACAGGTGTGAGCCACTGTGCCCA
CTAGGATTACAGGTGTGAGCCACTGTGCCCAG
TAGGATTACAGGTGTGAGCCACTGTGCCCAGC
AGGATTACAGGTGTGAGCCACTGTGCCCAGCC
GGATTACAGGTGTGAGCCACTGTGCCCAGCCA
GATTACAGGTGTGAGCCACTGTGCCCAGCCAA
ATTACAGGTGTGAGCCACTGTGCCCAGCCAAG
ACTGTGCCCAGCCAAGAAAAACTTCTAAAGTT
GATGTTTCTTTAACCAGCCAGATGCAAATAAC
```

For 1M last 20 fragments:

The program is still running for the 1M 32-character long fragments. Due to this reason, I do not have the last 20 fragments for the 1M character long fragments.

B. (30 pts)

1: How long did it take you to search for the first 10K, 100K, and 1M 32-character long fragments of the subject dataset within the query dataset?

Ans:

Time taken to search for the first 10K: 0.074871 seconds.

Time taken to search for the first 100K: 0.625749 seconds.

Time taken to search for the first 1M: 4.95256 seconds.

2: How long would it take to search for every possible 32-character long fragment of the subject dataset within the query dataset? Please note that depending on the efficiency of your algorithm, this step may take a long time. If the total time estimate is greater than 24 CPU hours, provide an estimate rather than exact number.

Total time taken is: 13519.3 seconds and it is approximately 3.7552778 hours.

← → ↺ 🏠 🔍 ondemand.hpc.nau.edu/pun/sys/dashboard/files/fs//home/ps747/Homework2/Isds\_hw2\_output\_B5

```
AGCCCAACAAGATTCCAATAGATGTGGTGTTA
time taken for first 1 million: 4.95256 sec
fragments found in first million
ATCCTGAGGGCCAGGTGCAGTGGCTCACGCCT
AGGGCCAGGTGCAGTGGCTCACGCCTGTAATC
GCCAGGTGCAGTGGCTCACGCCTGTAATCACA
GTGCAGTGGCTCACGCCTGTAATCACAGCACT
TGCAGTGGCTCACGCCTGTAATCACAGCACTT
GCAGTGGCTCACGCCTGTAATCACAGCACTTT
CAGTGGCTCACGCCTGTAATCACAGCACTTTG
AGTGGCTCACGCCTGTAATCACAGCACTTTGG
GCCCAGGAGTTTGAGACCAACCTGGGCAACAT
CCCAGGAGTTTGAGACCAACCTGGGCAACATG
AGGAGTTTGAGACCAACCTGGGCAACATGGCA
ATTAGCCGGGTGTGGTGGCATGTGCCTGTAGT
AGCCGGGTGTGGTGGCATGTGCCTGTAGTTCC
CCGGGTGTGGTGGCATGTGCCTGTAGTTCCAG
GGTGGGAGGATCACTTGACCCTAGGAGGACAA
CTCCAGCCTGGACGACAGAGTGAGACTCTGTC
TCCAGCCTGGACGACAGAGTGAGACTCTGTCT
CCAGCCTGGACGACAGAGTGAGACTCTGTCTC
CAGCCTGGACGACAGAGTGAGACTCTGTCTCA
GCCTGGACGACAGAGTGAGACTCTGTCTCAA
total time taken: 13519.3 sec
```

---

3: Print the 20 query fragments found within the subject dataset that have the largest indices (i.e. found later in the subject) for the first 10K, 100K, and 1M 32-character long fragments.

Ans:

Below is the screenshot for the 20 query fragments found within the first 10K.

```
← → ↺ 🏠 🔍 ondemand.hpc.nau.edu/pun/sys/dashboard/files/fs//home/ps747/Homework2/lsds_hw2_output_B5

queries filesize:5188888898

queries read successfully: 100000000
gnome size3095411481
total chars read from gnome file:3057186663
started sorting
Queries Sorted

Question: B
time taken for first 10k: 0.074871 sec
fragments found in first 10k
CCAAACCCAATCCCAGCAAACAAAGGAAATAA
AACCCAATCCCAGCAAACAAAGGAAATAACCA
CAATCCCAGCAAACAAAGGAAATAACCAAGAT
CCAGCAAACAAAGGAAATAACCAAGATCAGAG
AACAAAGGAAATAACCAAGATCAGAGCAGAAC
AAAGGAAATAACCAAGATCAGAGCAGAACTAA
AAGGAAATAACCAAGATCAGAGCAGAACTAAA
AGGAAATAACCAAGATCAGAGCAGAACTAAAT
GGAAATAACCAAGATCAGAGCAGAACTAAATG
GAAATAACCAAGATCAGAGCAGAACTAAATGA
AAATAACCAAGATCAGAGCAGAACTAAATGAA
AATAACCAAGATCAGAGCAGAACTAAATGAAA
ATAACCAAGATCAGAGCAGAACTAAATGAAAT
TAACCAAGATCAGAGCAGAACTAAATGAAATT
AACCAAGATCAGAGCAGAACTAAATGAAATTG
ACCAAGATCAGAGCAGAACTAAATGAAATTGA
AAGATCAGAGCAGAACTAAATGAAATTGACAC
AGATCAGAGCAGAACTAAATGAAATTGACACA
CTAAATGAAATTGACACAACAACAACAAC
AAACATAAATAAAACAAAAATTTGGTTATTTG
```



Below is the screenshot for the 20 query fragments found within the first 100K.

```
ondemand.hpc.nau.edu/pun/sys/dashboard/files/fs//home/ps747/Homework2/lsts_hw2_output_B5
time taken for first 100k: 0.625749 sec
fragments found in first 100k
AGTGCTAGGATTACAGGTGTGAGCCACTGTGC
GTGCTAGGATTACAGGTGTGAGCCACTGTGCC
TGCTAGGATTACAGGTGTGAGCCACTGTGCCC
GCTAGGATTACAGGTGTGAGCCACTGTGCCCC
CTAGGATTACAGGTGTGAGCCACTGTGCCCCAG
TAGGATTACAGGTGTGAGCCACTGTGCCCCAGC
AGGATTACAGGTGTGAGCCACTGTGCCCCAGCC
GGATTACAGGTGTGAGCCACTGTGCCCCAGCCA
GATTACAGGTGTGAGCCACTGTGCCCCAGCCAA
ATTACAGGTGTGAGCCACTGTGCCCCAGCCAAG
ACTGTGCCCCAGCCAAGAAAAAATTCTAAAGTT
GATGTTTCTTTAACCAGCCAGATGCAAATAAC
ACTACCCAACATGGTAGCCAGTGGTCACTAAT
GCAGATACAGGACATTCCCGTTATCTCAGAGA
GAAAAAGAGGAGCTAGGTGGATGAGGAGTGTA
CTCAGTGGCCATATTTCTCAATGTGCAGCCA
TTTCTCTCTTACAGAGTTTGAAATGCTGCTTG
TGATTGAATGATGATGCCTCACTTTACATAC
TGAATGATGATGCCTCACTTTACATACAGAT
AGCCCAACAAGATTCCAATAGATGTGGTGTTA
```

Below is the screenshot for the 20 query fragments found within the first 1M.

```
ondemand.hpc.nau.edu/pun/sys/dashboard/files/fs//home/ps747/Homework2/lsts_hw2_output_B5
AGCCCAACAAGATTCCAATAGATGTGGTGTTA
time taken for first 1 million: 4.95256 sec
fragments found in first million
ATCCTGAGGGCCAGGTGCAGTGGCTCACGCCT
AGGGCCAGGTGCAGTGGCTCACGCCTGTAATC
GCCAGGTGCAGTGGCTCACGCCTGTAATCACA
GTGCAGTGGCTCACGCCTGTAATCACAGCACT
TGCAGTGGCTCACGCCTGTAATCACAGCACTT
GCAGTGGCTCACGCCTGTAATCACAGCACTTT
CAGTGGCTCACGCCTGTAATCACAGCACTTTG
AGTGGCTCACGCCTGTAATCACAGCACTTTGG
GCCCAGGAGTTTGAGACCAACCTGGGCAACAT
CCCAGGAGTTTGAGACCAACCTGGGCAACATG
AGGAGTTTGAGACCAACCTGGGCAACATGGCA
ATTAGCCGGGTGTGGTGGCATGTGCCTGTAGT
AGCCGGGTGTGGTGGCATGTGCCTGTAGTTCC
CCGGGTGTGGTGGCATGTGCCTGTAGTTCCAG
GGTGGGAGGATCACTTGACCCTAGGAGGACAA
CTCCAGCCTGGACGACAGAGTGAGACTCTGTC
TCCAGCCTGGACGACAGAGTGAGACTCTGTCT
CCAGCCTGGACGACAGAGTGAGACTCTGTCTC
CAGCCTGGACGACAGAGTGAGACTCTGTCTCA
GCCTGGACGACAGAGTGAGACTCTGTCTCAA
total time taken: 13519.3 sec
```