```
//Fernando Felix
//Files Associated: Assign1.cpp
//This program maintains two data structures that hold a random string of characters
ranging from
//1mb to 2mb. One structure named recent list is searched for a String and keeps
track of how
//many instances this string is a substring of the words in the array. It is then
ejected and updated with strings from library
using namespace std;
#include <iostream>
#include <stdlib.h>
#include <string>
#include <cstring>
#include <ostream>
#include <deque>
#include<cmath>

//user choice of strings
string getString(int x);

//Struct of pointer
struct pointer
{
        string * p;

};

//main program
int main()
{
        //initialize variables
        int librarySize = 1024;  //variable to hold the library size
        int list = 128;         //variable to hold list size
        long maxByte=2000000;  //2 mb
        long minByte=1000000; //1mb
        int randomChar;              //random character to be appended to string
        int instance=0;              //instances of the string found
        string searchString;         //string to search for
        double randomByte;           //to hold range of 1-2mb

        //create arrays
        deque<string> library; //holds 1024 strings of up to 2mb
        struct pointer recent_list;   //array of pointers to recent list
        recent_list.p=new string[128];




        //initialize list to random strings
        for (int x=0;x<list;x++)
        {
                randomByte=rand()%(maxByte-minByte)+minByte; //1mb-2mb
                string s;
        for (int j = 0; j<randomByte; j++)
          {
                randomChar= rand()% (90 - 65) + 65; //Character A-Z
                s += (char)randomChar;              //append char to a string

           }
                (recent_list.p[x])=s;                        //insert string to array
        }
        //initialize library

        for (int x=0;x<librarySize;x++)
        {
                randomByte=rand()%(maxByte-minByte)+minByte; //1mb-2mb
                string s;
        for (int j = 0; j<randomByte; j++)
          {
                randomChar= rand()% (90 - 65) + 65; //Character A-Z
                s += (char)randomChar;

           }
                library.push_back(s);
        }



                //ask user for string to search
                cout <<"Enter a string to look for"<<endl<<"1.FIRST 2.CPP 3.REVIEW"<<
                "4.PROGRAM 5.ASSIGNMENT 6.CECS 7.BEACH 8.ECS 9.FALL 10.SPRING 11.OS"
<<
                 "12.MAC  13.LINUX 14.WINDOWS 15. LAB."<<endl<<"Enter 0 to
exit"<<endl;
                int choice;
                cin>>choice;
                searchString=getString(choice);

                cout<<"String Searched: "<<searchString<<endl;
        //loop to ask user for string input to search in recent list
        do
        {

        //search for the string in array, set to empty if not found. keep same if
found
        for(int x=0; x<list ;x++)
          {

                size_t find = recent_list.p[x].find(searchString);
                if(find!=string::npos )
                {
                        instance++;

                }
                else
                {
                        string current = recent_list.p[x];
                        library.push_back(current);
                        recent_list.p[x]="empty";

                }



        }//for loop

        //shift and reinitialize
        int y=list-1;
        for(int x=0; x<floor(list/2); x++)
        {
           if(recent_list.p[x]=="empty" && recent_list.p[x]!="empty")
           {

                recent_list.p[x]=recent_list.p[y];
                recent_list.p[y]="empty";

           }
           else if((recent_list.p[x]!="empty") && (recent_list.p[y]=="empty"))
                {  int z=x;
                   while(recent_list.p[z]!="empty")
                   {z++;
                   if(&(recent_list.p[x])==(&recent_list.p[z]))
                        break;
                    }
                   recent_list.p[z]=recent_list.p[y];
                   recent_list.p[y]="empty";

                }
           else if(recent_list.p[x]=="empty" && recent_list.p[y]=="empty")
                {
                   int z=y;
                   while(recent_list.p[z]=="empty")
                     {z--;
                        if((recent_list.p[x])==(recent_list.p[z]))
                        break;
                    }
                      recent_list.p[x]=recent_list.p[z];
                      recent_list.p[z]="empty";

                }

                y--;
        }

        //insert from library
        for(int x = 0; x<list-instance; x++)
        {
                recent_list.p[x+instance]=library.front();
                library.pop_front();
        }


        cout<<searchString <<": "<<list-instance<< " documents ejected &
reinitialized" <<endl;
        instance=0;

        cout <<"Enter a string to look for"<<endl<<"1.FIRST 2.CPP 3.REVIEW"<<
        "4.PROGRAM 5.ASSIGNMENT 6.CECS 7.BEACH 8.ECS 9.FALL 10.SPRING 11.OS" <<
        "12.MAC  13.LINUX 14.WINDOWS 15. LAB."<<endl<<"Enter 0 to exit"<<endl;
        cin>>choice;
        searchString=getString(choice);



}while(searchString!="0");
delete recent_list.p;

}

//contains a dictionary of strings to search for
string getString(int x)
{
        switch(x)
        {
        case 1:
        return "FIRST";
        break;
        case 2:
        return "CPP";
        break;
        case 3:
        return "REVIEW";
        break;
        case 4:
        return "PROGRAM";
        break;
        case 5:
        return "ASSIGNMENT";
        break;
        case 6:
        return "CECS";
        break;
        case 7:
        return "BEACH";
        break;
        case 8:
        return "ECS";
        break;
        case 9:
        return "FALL";
        break;
        case 10:
        return "SPRING";
        break;
        case 11:
        return "OS";
        break;
        case 12:
        return "MAC";
        break;
        case 13:
        return "LINUX";
        break;
        case 14:
        return "WINDOWS";
        break;
        case 15:
        return "LAB";
        break;
        }

        return "0";

}
```