```
#include<iostream>
using namespace std;
#include<string>

typedef int BOOL;
#define TRUE 1
#define FALSE 0

class String
{
protected:
    char * arr;
    int capacity;
    int length;

public:
    String();//default const
    String(char *);//paramtersied const by char string
    String(int);////paramtersied const by int
    String(String &);//copy constructor
    ~String();//Destructor

    void setString(char *);//method used if we inaitlised string with size

    int GetCapacity();//returns total capacity
    int GetLength();//return length("\0" included)
    char CharAt(int);//returns a char value at the given index number

    void concat(char *);//combines specified string at the end of this string.
                        //It returns combined string

    void Display();//Displays string
    void trim();//method eliminates leading and trailing spaces

    int contains(char *);  //conatains function in java

    String* substring(int, int);////return all the characters from startIndex
      to endIndex
    String* substring(int);//return all the characters from startIndex

    void toLowerCase();//returns the string in lowercase letter
    void toUpperCase();//returns the string in uppercase letter
    void FirstCap();//returns the string in uppercase Fisrt letter of each Word

    void increseCapacity(int);

    BOOL equals(String);//method compares the two given strings based on the
      content of the string. If any character is not matched, it returns false.
```

```
47                          //If all characters are matched, it returns true
48
49      int compareTo(String);/*if s1 > s2, it returns positive number
50                              if  s1 < s2, it returns negative number
51                              if S1 == s2, it returns 0
52                              String s1="hello";
53                              String s2="hello";
54                              String s3="meklo";
55                              String s4="hemlo";
56                              String s5="flag"
57                              s1.compareTo(s2));//0 because both are equal
58                              s1.compareTo(s3));//-5 because "h" is 5 times lower
                        than "m"
59                              s1.compareTo(s4));//-1 because "l" is 1 times lower
                        than "m"
60                              s1.compareTo(s5));//2 because "h" is 2 times
                       greater than "f"  */
61
62      const char* replace(char old, char newc);//replaces only single character
          in String
63      const char* replaceFirst(char* old, char* newc);//replaces only first
          instances of char_String
64      const char* replaceAll(char* old, char* newc);//replaces all instances of
          char_String
65
66      char** split(char ,int);// method splits this string against given regular
          expression
67                              //and returns a char array
68
69
70  };
71
72
73  String::String()
74  {
75      capacity = 10;
76      length = 0;
77      arr = new char[capacity];
78  }
79
80  String::String(char *str)
81  {
82
83      int i = 0;
84      for (i = 0;str[i] != '\0';i++);
85      length=capacity =i + 1;
86
87      this->arr = new char[capacity];
88
```

```
 89      for (i = 0;str[i] != '\0';i++)
 90      {
 91
 92          this->arr[i] = str[i];
 93      }
 94      this->arr[i] =(char)'\0';
 95  }
 96
 97  String::String(int size)
 98  {
 99      capacity = size;
100      length = 0;
101      arr = new char[capacity];
102
103  }
104
105  String::String(String &ref)
106  {                                       //to copy old object into new object
107
108      if(ref.capacity != 0)
109      {
110          this->capacity = ref.capacity;          //deep copy
111          this->length = ref.length;
112          this->arr = new char[capacity];
113          int i = 0;
114          for (i = 0;ref.arr[i] != '\0';i++)
115          {
116              this->arr[i] = ref.arr[i];
117          }
118          this->arr[i] = (char)'\0';
119      }
120  }
121
122  String::~String()
123  {                               //destrctor
124      if (arr != NULL)
125      {
126          delete[]arr;
127      }
128
129  }
130
131  char String::CharAt(int index)
132  {
133      return (this->arr[index]);////((char)this->arr[index]);
134  }
135
136  int String::GetCapacity()
137  {
```

```
138     return this->capacity;
139 }
140
141 int String::GetLength()
142 {
143     return this->length;
144 }
145
146 void String::Display()
147 {
148     if (this->arr != NULL)
149     {
150         cout << this->arr << endl;
151     }
152
153 }
154
155 void String::concat(char * str)
156 {
157     int i = 0, size = 0,j=0;
158     for (i = 0;str[i] != '\0';i++, size++);
159
160     int iRet =capacity-length;
161
162
163     if (size < iRet)
164     {
165         for (i = 0, j = length - 1;str[i] != '\0';j++, i++)
166         {
167
168             this->arr[j] = str[i];
169         }
170         this->arr[j] = (char)'\0';
171         this->length=length+size;
172     }
173     else
174     {
175
176         char * temp = new char[length];
177         for (i = 0;this->arr[i] != '\0';i++)
178         {
179             temp[i] = this->arr[i];         //copy old string into temprory
                  string
180         }
181         temp[i] = '\0';
182
183         delete[]arr;
184         this->arr = new char[length + size];//allocate new size
185
```

```
186         for (i = 0;i <temp[i]!='\0';i++)
187         {
188             this->arr[i] = temp[i];     //copy old data into new memory
189         }
190         this->arr[i] = '\0';
191 
192         for (j=0;str[j] != '\0';j++,i++)
193         {
194             this->arr[i] = str[j];        //concat new string to old data in
                  new memory
195         }
196         this->arr[i] = '\0';
197 
198         //this->capacity = length + size;
199         //this->length = length + size;
200 
201     }
202 
203 }
204 
205 void String::setString(char *str)
206 {
207     int i = 0;
208     for (i = 0;(i < this->capacity) && (str[i] != '\0');i++)
209     {
210         this->arr[i]=str[i];
211     }
212     this->arr[i]='\0';
213 }
214 void String::trim()
215 {
216     int i = 0,j=0;
217     char temp = '\0';
218 
219     for(i = 0,j=0;arr[i] != '\0';i++)
220     {
221 
222         if (arr[i] !=' ')
223         {
224             arr[j] = arr[i];
225 
226             j++;
227         }
228     }
229     this->arr[j] = '\0';
230 
231 }
232 
233 // strstr = contains
```

```
234 int String::contains(char * str)
235 {
236     int i = 0, j = 0;
237     int pos = -1;
238     int flag = 0;
239 
240     for (i = 0;this->arr[i] != '\0';i++)
241     {
242         if (arr[i] == str[j])
243         {
244             pos = i;
245             int l = i+1;
246             if (str[j + 1] == '\0')
247             {
248                 flag = 1;
249             }
250             for (j++;(str[j] != '\0') && (arr[l] != '\0') ;j++)
251             {
252                     if (arr[l] == str[j])
253                     {
254                        flag = 1;
255                         l++;
256                     }
257                     else if (flag != 1)
258                     {
259                         break;
260                     }
261                     else
262                     {
263                         flag = 0;
264                     }
265             }
266 
267         }
268         j = 0;
269 
270         if (flag == 1)
271         {
272             break;
273         }
274         //flag = 0;
275     }
276 
277     if ((pos!=-1) && (flag == 1))
278     {
279         return pos;
280     }
281     else
282     {
```

```
283         return -1;
284     }
285 }
286
287 String* String::substring(int begin, int end)
288 {
289     if ((begin < 0) || (begin > this->length)|| (end < 0)|| (begin > end))
290     {
291         return NULL;
292     }
293
294             int iRet = ((end - begin) + 1);
295             String* AnsStr = new String(iRet);
296             int i = 0, j = 0;
297
298             for (i = begin;(i < end) && (this->arr[i] != '\0');i++, j++)
299             {
300                 AnsStr->arr[j] = this->arr[i];
301
302             }
303
304             AnsStr->arr[j] = '\0';
305
306             AnsStr->length = AnsStr->capacity = iRet;
307             return AnsStr;
308
309 }
310
311 String* String::substring(int begin)
312 {
313     if ((begin < 0) || (begin > this->length))
314     {
315         return NULL;
316     }
317
318     int iLength = ((this->length)-begin);
319
320     String* AnsStr = new String(iLength);
321     int j = 0;
322     for (int i = begin;this->arr[i] != '\0';i++, j++)
323     {
324         AnsStr->arr[j] = this->arr[i];
325     }
326     AnsStr->arr[j] = '\0';
327
328     AnsStr->length = AnsStr->capacity = iLength;
329
330     return AnsStr;
331 }
```

```
332
333 void String::toLowerCase()
334 {
335     if (this->arr != NULL)
336     {
337         for (int i = 0;this->arr[i] != '\0';i++)
338         {
339             if ((this->arr[i] >= 'A') && (this->arr[i] <= 'Z'))
340             {
341                 this->arr[i]=(this->arr[i]+32);
342             }
343         }
344     }
345 }
346
347 void String::toUpperCase()
348 {
349     if (this->arr != NULL)
350     {
351         for (int i = 0;this->arr[i] != '\0';i++)
352         {
353             if ((this->arr[i] >= 'a') && (this->arr[i] <= 'z'))
354             {
355                 this->arr[i] = (this->arr[i] - 32);
356             }
357         }
358     }
359 }
360
361 void String::FirstCap()  // all first letters of words in string makes capital
362 {
363     int i=0;
364
365     if ((this->arr[i] >= 'a') && (this->arr[i] <= 'z'))
366     {
367         this->arr[i] = (this->arr[i] - 32);
368     }
369     for (i = 1;this->arr[i] != '\0';i++)
370     {
371         if (this->arr[i - 1] == ' ')
372         {
373             if ((this->arr[i] >= 'a') && (this->arr[i] <= 'z'))
374             {
375                 this->arr[i] = (this->arr[i] - 32);
376             }
377         }
378     }
379 }
380
```

```
381 void String::increseCapacity(int size)
382 {
383     char * newArr = new char[size];
384     int i = 0;
385 
386     for (i = 0; i < size;i++)
387     {
388         if (arr[i] == '\0')
389         {
390             break;
391         }
392         newArr[i] = this->arr[i];
393 
394     }
395     newArr[i] = '\0';
396 
397     this->arr = newArr;
398 
399     this->capacity = size;
400 }
401 
402 
403 int String::compareTo(String str)
404 {
405     if ((this->arr == NULL) && (str.arr == NULL))
406     {
407         return 0;
408     }
409     if (this->length < str.length)
410     {
411         return -1;// returns -1 if arugument String is smaller
412     }
413     if (this->length > str.length)
414     {
415         return 1;// returns +1 if arugument String is Greater
416     }
417     int i = 0;
418     int diff = 0;
419     if (this->length == str.length)
420     {
421 for (i = 0; ((this->arr[i] != '\0') && (str.arr[i] != '\0')); i++)
422 {
423     if (this->arr[i] != str.arr[i])
424     {
425         diff = this->arr[i] - str.arr[i];
426         break;
427     }
428 }
429 if (diff != 0)
```

```
430 {
431     return diff;//
432 }
433 else
434 {
435     return 0;// returns 0 if arugument String is equal
436 }
437     }
438
439     return 0;
440 }
441
442 const char* String::replace(char old, char newc)
443 {
444     int i = 0;
445     for (i = 0; i < this->length; i++)
446     {
447         if (this->arr[i] == old)
448         {
449             this->arr[i] = newc;
450         }
451     }
452     return this->arr;
453 }
454
455 const char* String::replaceFirst(char* old, char* newc)
456 {
457     int iRet = this->contains(old);
458
459     if (iRet == -1)
460     {
461         return NULL;
462     }
463     int ol = 0, nl = 0, m = 0;
464     while (1)
465     {
466         if (old[ol] != '\0')
467         {
468             ol++;
469         }
470         if (newc[nl] != '\0')
471         {
472             nl++;
473         }
474         if ((newc[nl] == '\0') && (old[ol] == '\0'))
475         {
476             break;
477         }
478     }
```

```
479     String* temp = new String(this->arr);
480
481     length = (length + (nl - ol));
482     arr = new char[length];
483     int i = 0, j = 0;
484
485     for (i = 0, j = 0; i < iRet; j++, i++)
486     {
487         this->arr[i] = temp->arr[j];
488     }
489     for (m = 0; m < nl; i++, m++)
490     {
491         this->arr[i] = newc[m];
492
493     }
494     for (j = j + ol; temp->arr[j] != '\0'; j++, i++)
495     {
496         this->arr[i] = temp->arr[j];
497     }
498     this->arr[i] = '\0';
499     delete[]temp->arr;
500     return arr;
501 }
502
503 const char* String::replaceAll(char* old, char* newc)
504 {
505     while (1)
506     {
507         if (this->replaceFirst(old, newc) != NULL)
508         {
509         }
510         else
511         {
512             break;
513         }
514
515     }
516     return this->arr;
517 }
518
519 char** String::split(char str, int limit)
520 {
521     if (limit < 0)
522     {
523         limit = -(limit);
524     }
525     char** chararr = new char*[limit];
526     for (int i = 0; i < limit; i++)
527     {
```

```
528             chararr[i] = new char[30];
529         }
530         int i = 0, j = 0, k = 0;
531         while (this->arr[i] != '\0')
532         {
533
534             if (arr[i] != str)
535             {
536                 chararr[j][k] = this->arr[i];
537                 k++;
538                 i++;
539             }
540             else
541             {
542                 chararr[j][k] = '\0';
543                 k = 0;
544                 i++;
545                 j++;
546             }
547
548         }
549         chararr[j][k] = '\0';
550         j++;
551         if ((arr[i] == '\0') && (j < limit))
552         {
553
554             for (; j < limit; j++)
555             {
556                 //cout << j << endl;
557                 chararr[j][0] = '\0';
558             }
559         }
560         return chararr;
561     }
562
563     BOOL String::equals(String str)
564     {
565         int i = 0;
566         if ((this->length == str.length)&&(this->arr!=NULL)&&(str.arr!=NULL))
567         {
568             for (i = 0; ((this->arr[i] != '\0') && (str.arr[i] != '\0')); i++)
569             {
570                 if (this->arr[i] != str.arr[i])
571                 {
572                         break;
573                 }
574             }
575             if (i<this->length)
576             {
```

```
577             return TRUE;// returns 0 if arugument String is equal
578         }
579         else
580         {
581             return FALSE;
582         }
583     }
584     else
585     {
586         return FALSE;
587     }
588 
589 
590 }
591 //main function**********************
592 
593 int main()
594 {
595     /*cout << "Enter String" << endl;
596     char str[30];
597     cin.getline(str, 30);
598     String S1(str);
599 
600         cout << "Enter Search String" << endl;
601         char str2[30];
602         cin >> str2;
603 */
604     String S1("shubham Dharma Rasal");
605     S1.Display();
606     cout << endl << endl;
607     cout << "Charater at method\tS1.CharAt(8)"<< endl<<endl;
608     cout << S1.CharAt(8)<<endl;
609 
610     cout << "Get Capacity method\tS1.GetCapacity()" << endl << endl;
611     cout << S1.GetCapacity()<<endl;
612 
613     cout << "Get Length method\tS1.GetLength()" << endl << endl;
614     cout << S1.GetLength()<<endl;
615 
616     cout << "concat method\tS1.concat(""13 / 05"")" << endl << endl;
617     S1.concat("13/05");
618     S1.Display();
619     cout << endl << endl;
620 
621     cout << "contains method\tS1.contains(""Dharma"")" << endl << endl;
622     int iRet = S1.contains("Dharma") + 1;//contains
623         if (iRet)
624         {
625             cout << "FOUND AT POS\t" << iRet << endl<<endl;
```

```
626         }
627         else
628         {
629             cout << " NOT FOUND!!!\n";
630         }
631         cout << "subString method\tS1.substring(3)" << endl << endl;
632         String * Substring = NULL;
633         Substring = S1.substring(3);
634         if (Substring != NULL)
635         {
636             Substring->Display();
637 
638             cout << endl << endl;
639         }
640         else
641         {
642             cout << "NOT FOUND" << endl;
643         }
644 
645         cout << "subString method\tS1.substring(4,11)" << endl << endl;
646         Substring = S1.substring(4,11);
647         if (Substring != NULL)
648         {
649             Substring->Display();
650 
651             cout << endl << endl;
652         }
653         else
654         {
655             cout << "NOT FOUND" << endl;
656         }
657         cout << "toLowerCase method\tS1.toLowerCase" << endl << endl;
658          S1.toLowerCase();
659          S1.Display();
660          cout << endl << endl;
661 
662          cout << "firstCap method\tS1.FirstCap" << endl << endl;
663          S1.FirstCap();
664          S1.Display();
665          cout << endl << endl;
666 
667          cout << "toUpperCase method\tS1.toUpperCase" << endl << endl;
668          S1.toUpperCase();
669          S1.Display();
670          cout << endl << endl;
671 
672          String S2("SHUBHAM DHARMA RASAL");
673          S2.Display();
674          cout << endl;
```

```
675
676             cout << "compareTo method\tS1.compareTo" << endl << endl;
677             cout << S1.compareTo(S2)<<endl<<endl;
678
679             cout << "replace(char) method\tS2.replace('A','B')" << endl << endl;
680             cout << S2.replace('A', 'B') << endl << endl;
681
682             cout << "replaceFirst(String) method\tS2.replaceFirst
                  (""SHUBHAM"",""shubham"")" << endl << endl;
683             cout << S2.replaceFirst("RBSBL", "RASAL")<<endl<<endl;
684
685             cout << "replaceAll(String) method\tS2.replaceAll(""HB"",""***"")" <<
                  endl << endl;
686             cout << S2.replaceAll("HB", "***") << endl << endl;
687
688             cout << "AGAIN replaceAll(String) method\tS2.replaceAll
                  (""***"",""HA"")" << endl << endl;
689             cout << S2.replaceAll("***", "HA") << endl << endl;
690
691
692             cout << "split(Stirng to array) method\tS2.split(' ',3)" << endl <<
                  endl;
693
694                char** ary=S2.split(' ',3);
695                cout << "printing an array:=" << endl;
696                for (int i = 0; i < 3; i++)
697                {
698                 for (int j = 0; ary[i][j] != '\0'; j++)
699                 {
700                    cout << ary[i][j];
701                 }
702                 cout << endl;
703                }
704
705                cout << "\ntrim method\tS2.trim()" << endl << endl;
706                S2.trim();
707                S2.Display();
708                cout<< endl << endl;
709
710                String S3("shubham");
711                String S4("shubham");
712
713                cout << S4.GetLength() << endl;
714                cout << "\nequals method\tS3.equals(S4)" << endl << endl;
715                if (S3.equals(S4) == TRUE)
716                {
717                    cout << "objects are Equals" << endl;
718                }
719                else
```

```
720             {
721                 cout << "objects are not equal";
722             }
723 
724 
725         return 0;
726 }
727 /*
728 OUTPUT:==
729 
730 shubham Dharma Rasal
731 
732 
733 Charater at method      S1.CharAt(8)
734 
735 D
736 Get Capacity method     S1.GetCapacity()
737 
738 21
739 Get Length method       S1.GetLength()
740 
741 21
742 concat method   S1.concat(13 / 05)
743 
744 shubham Dharma Rasal13/05
745 
746 
747 contains method S1.contains(Dharma)
748 
749 FOUND AT POS    9
750 
751 subString method        S1.substring(3)
752 
753 bham Dharma Rasal13/05
754 
755 
756 subString method        S1.substring(4,11)
757 
758 ham Dha
759 
760 
761 toLowerCase method      S1.toLowerCase
762 
763 shubham dharma rasal13/05
764 
765 
766 firstCap method S1.FirstCap
767 
768 Shubham Dharma Rasal13/05
```

```
769
770
771 toUpperCase method      S1.toUpperCase
772
773 SHUBHAM DHARMA RASAL13/05
774
775
776 SHUBHAM DHARMA RASAL
777
778 compareTo method        S1.compareTo
779
780 0
781
782 replace(char) method    S2.replace('A','B')
783
784 SHUBHBM DHBRMB RBSBL
785
786 replaceFirst(String) method     S2.replaceFirst(SHUBHAM,shubham)
787
788 SHUBHBM DHBRMB RASAL
789
790 replaceAll(String) method       S2.replaceAll(HB,***)
791
792 SHUB***M D***RMB RASAL
793
794 AGAIN replaceAll(String) method S2.replaceAll(***,HA)
795
796 SHUBHAM DHARMB RASAL
797
798 split(Stirng to array) method   S2.split(' ',3)
799
800 printing an array:=
801 SHUBHAM
802 DHARMB
803 RASAL
804
805 trim method     S2.trim()
806
807 SHUBHAMDHARMBRASAL
808
809
810 8
811
812 equals method   S3.equals(S4)
813
814 objects are Equals
815
816 E:\c++\String\x64\Debug\String.exe (process 8748) exited with code 0.(0 means
      success)
```

```
817 Press any key to close this window . . .
818
819 */
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
```