This is not everybody's cup of tea.

Introspect yourself.

You are the best judge of your own standard and you can fool others around you but not yourself.

Don't waste time on this document if you are not up to it. Everybody cannot become a programmer.

Coding
Science
Product Development

```
//Example 1
#include<stdio.h>
int lmn();
int pqr();
int  main()
{
printf("%d %d\n",lmn(),pqr());
return 0;
}
int lmn()
{
printf("One\n");
return 1;
}
int pqr()
{
printf("two\n");
return 2;
}
```

```
//Example 2
#include<stdio.h>
int main()
{
int x;
x=10;
printf("%d",x=x+50);
return 0;
}
```

```
//Example 3
#include<stdio.h>
int main()
{
if(1)
{
printf("Hello\n");
}
if(0)
{
printf("God\n");
}
if(2)
{
printf("Great\n");
}
if(-2)
{
```

```
printf("Blue\n");
}
if(-0)
{
printf("Ujjain");
}
return 0;
}
```

```
//Example 4
#include<stdio.h>
int main()
{
int x;
x=1;
if(--x)
{
printf("Great");
}
printf("%d\n",x);
return 0;
}
```

```
//Example 5
#include<stdio.h>
void main()
{
int x;
x=1;
if(x--)
{
printf("Great\n");
}
printf("%d\n",x);
return 0;
}
```

```
//Example 6
#include<stdio.h>
int main()
{
int x;
x=1;
if(x-- && x--)
{
printf("Great");
}
printf("%d\n",x);
return 0;
```

```
}
```

```
//Example 7
#include<stdio.h>
int main()
{
int x;
x=1;
if(x-- && ++x)
{
printf("Great");
}
printf("%d\n",x);
return 0;
}
```

```
//Example 8
#include<stdio.h>
int main()
{
int x;
x=1;
if(--x && ++x)
{
printf("Great");
}
printf("%d\n",x);
return 0;
}
```

```
//Example 9
#include<stdio.h>
int main()
{
int x;
x=1;
if(--x && x++)
{
printf("Great");
}
printf("%d\n",x);
return 0;
}
```

```
//Example 10
#include<stdio.h>
int main()
{
int x;
x=1;
```

```
if(--x || x++)
{
printf("Great");
}
printf("%d\n",x);
return 0;
}
```

```
//Example 11
#include<stdio.h>
int main()
{
int x;
x=1;
if(--x || x++ || ++x || ++x || --x || ++x)
{
printf("Great");
}
printf("%d\n",x);
return 0;
}
```

```
//Example 12
#include<stdio.h>
int main()
{
int x;
x=10>20;
printf("%d",x);
return 0;
}
```

```
//Example 13
#include<stdio.h>
int main()
{
int x;
x=10<20;
printf("%d",x);
return 0;
}
```

```
//Example 14
#include<stdio.h>
int main()
{
int x;
x=10<20 && 30<100;
printf("%d",x);
return 0;
```

```
}
```

```
//Example 15
#include<stdio.h>
int main()
{
int x;
x=1 && 0;
printf("%d",x);
return 0;
}
```

```
//Example 16
#include<stdio.h>
int main()
{
int x;
x=10 || 20;
printf("%d",x);
return 0;
}
```

```
//Example 17
#include<stdio.h>
int main()
{
int x;
x=1 && 1 || 0;
printf("%d",x);
return 0;
}
```

```
//Example 18
#include<stdio.h>
int main()
{
int x;
x=1 || 1 && 0;
printf("%d",x);
return 0;
}
```

```
//Example 19
#include<stdio.h>
int main()
{
int x;
x=(1 || 1) && 0;
printf("%d",x);
return 0;
}
```

```
//Example 20
#include<stdio.h>
int main()
{
int x;
x=printf("Hello\n");
printf("%d",x);
return 0;
}
```

```
//Example 21
#include<stdio.h>
int main()
{
int x;
x=printf("%d %d %d",10,20,30);
printf("\n%d\n",x);
x=printf("%d%d%d",10,20,30);
printf("\n%d",x);
return 0;
}
```

```
//Example 22
#include<stdio.h>
int main()
{
int x,y,z;
x=10;
y=20;
z=30;
printf("%d %d %d",printf("%d",x),printf("%d",y),printf("%d",z));
return 0;
}
```

```
//Example 23
#include<stdio.h>
int lmn();
int pqr();
int main()
{
int x;
x=lmn() && pqr();
printf("%d",x);
return 0;
}
int lmn()
{
printf("left\n");
return 0;
```

```
}
int pqr()
{
printf("right\n");
return 0;
}
```

```
//Example 24
#include<stdio.h>
int lmn();
int pqr();
int main()
{
int x;
x=lmn() && pqr();
printf("%d",x);
return 0;
}
int lmn()
{
printf("left\n");
return 1;
}
int pqr()
{
printf("right\n");
return 0;
}
```

```
//Example 25
#include<stdio.h>
int lmn();
int pqr();
int main()
{
int x;
x=lmn() && pqr();
printf("%d",x);
return 0;
}
int lmn()
{
printf("left\n");
return 1;
}
int pqr()
{
printf("right\n");
```

```
return 1;
}
```

```
//Example 26
#include<stdio.h>
int lmn();
int pqr();
int main()
{
int x;
x=lmn() || pqr();
printf("%d",x);
return 0;
}
int lmn()
{
printf("left\n");
return 1;
}
int pqr()
{
printf("right\n");
return 1;
}
```

```
//Example 27
#include<stdio.h>
int lmn();
int pqr();
int main()
{
int x;
x=lmn() || pqr();
printf("%d",x);
return 0;
}
int lmn()
{
printf("left\n");
return 0;
}
int pqr()
{
printf("right\n");
return 1;
}
```

```
//Example 28
#include<stdio.h>
```

```
int lmn();
int pqr();
int rst();
int main()
{
int x;
x=lmn() || (pqr() && rst());
printf("%d",x);
return 0;
}
int lmn()
{
printf("left\n");
return 0;
}
int pqr()
{
printf("middle\n");
return 0;
}
int rst()
{
printf("right\n");
return 0;
}
```

```
//Example 29
#include<stdio.h>
int lmn();
int pqr();
int rst();
int main()
{
int x;
x=lmn() || (pqr() && rst());
printf("%d",x);
return 0;
}
int lmn()
{
printf("left\n");
return 0;
}
int pqr()
{
printf("middle\n");
return 0;
```

```
}
int rst()
{
printf("right\n");
return 0;
}
```

```
//Example 30
#include<stdio.h>
int lmn();
int pqr();
int rst();
int main()
{
int x;
x=(lmn() || pqr()) && rst();
printf("%d",x);
return 0;
}
int lmn()
{
printf("left\n");
return 0;
}
int pqr()
{
printf("middle\n");
return 0;
}
int rst()
{
printf("right\n");
return 0;
}
```

```
//Examle 31
#include<stdio.h>
int lmn();
int pqr();
int rst();
int main()
{
int x;
x=(lmn() || pqr()) && rst();
printf("%d",x);
return 0;
}
int lmn()
```

```
{
printf("left\n");
return 1;
}
int pqr()
{
printf("middle\n");
return 0;
}
int rst()
{
printf("right\n");
return 0;
}
```

```
//Example 32
#include<stdio.h>
int lmn();
int pqr();
int rst();
int main()
{
int x;
x=(lmn() || pqr()) && rst();
printf("%d",x);
return 0;
}
int lmn()
{
printf("left\n");
return 0;
}
int pqr()
{
printf("middle\n");
return 1;
}
int rst()
{
printf("right\n");
return 0;
}
```

```
//Example 33
#include<stdio.h>
int lmn();
int pqr();
int rst();
```

```
int main()
{
int x;
x=lmn() || (pqr() && rst());
printf("%d",x);
return 0;
}
int lmn()
{
printf("left\n");
return 0;
}
int pqr()
{
printf("middle\n");
return 1;
}
int rst()
{
printf("right\n");
return 0;
}
```

```
//Example 34
#include<stdio.h>
int main()
{
int x;
x=1;
if(--x)
{
printf("Great\n");
}
printf("%d\n",x);
return 0;
}
```

```
//Example 35
#include<stdio.h>
void main()
{
int x;
x=1;
if(x--)
{
printf("Great\n");
}
printf("%d\n",x);
```

```
return 0;
}
```

```
//Example 36
#include<stdio.h>
int main()
{
int x;
x=1;
if(x-- && x--)
{
printf("Great\n");
}
printf("%d\n",x);
return 0;
}
```

```
//Example 36
#include<stdio.h>
int main()
{
int x;
x=1;
if(x-- && ++x)
{
printf("Great\n");
}
printf("%d\n",x);
return 0;
}
```

```
//Example 37
#include<stdio.h>
int main()
{
int x;
x=1;
if(--x && ++x)
{
printf("Great\n");
}
printf("%d\n",x);
return 0;
}
```

```
//Example 38
#include<stdio.h>
int lmn();
int pqr();
int rst();
```

```
#include<stdio.h>
int main()
{
int x;
x=1;
if(--x && x++)
{
printf("Great\n");
}
printf("%d\n",x);
return 0;
}
```

```
//Example 39
#include<stdio.h>
int lmn();
int pqr();
int rst();
#include<stdio.h>
int main()
{
int x;
x=1;
if(--x || x++ || ++x || --x || ++x || ++x || ++x || ++x || ++x)
{
printf("Great\n");
}
printf("%d\n",x);
return 0;
}
```

```
//Example 40
#include<stdio.h>
int main()
{
int x;
x=1;
if(--x || x++ || ++x || --x || ++x || ++x || ++x || ++x || ++x)
{
printf("Great\n");
}
printf("%d\n",x);
return 0;
}
```

```
//Example 41
#include<Stdio.h>
int main()
{
```

```
int x,y,z,p;
x=10;
y=10;
z=10;
p=++x || (++y && ++z);
printf("%d %d %d\n",x,y,z);
printf("%d\n",p);
return 0;
}
```

```
//Example 42
#include<Stdio.h>
int main()
{
int x,y,z,p;
x=10;
y=10;
z=10;
p=(++x || ++y) && ++z;
printf("%d %d %d\n",x,y,z);
printf("%d\n",p);
return 0;
}
```

```
//Example 43
#include<Stdio.h>
int main()
{
int p;
p=10+20*3;
printf("%d\n",p);
return 0;
}
```

```
//Example 44
#include<Stdio.h>
int main()
{
int p;
p=(10+20)*3;
printf("%d\n",p);
return 0;
}
```

```
//Example 45
#include<Stdio.h>
int main()
{
int p;
p=10+20*3/3+1;
```

```
printf("%d\n",p);
return 0;
}
```

```
//Example 46
#include<Stdio.h>
int main()
{
int p;
p=10+20*3/3/10+1;
printf("%d\n",p);
return 0;
}
```

```
//Example 47
#include<Stdio.h>
int main()
{
int p;
p=10-20*3/3/10+1;
printf("%d\n",p);
return 0;
}
```

```
//Example 48
#include<stdio.h>
int main()
{
int x;
x=2147483648;
printf("%d",x);
return 0;
}
```

```
//Example 49
#include<stdio.h>
int main()
{
int x;
x=2147483649;
printf("%d",x);
return 0;
}
```

```
//Example 50
#include<stdio.h>
int main()
{
int x;
x=2147483650;
printf("%d",x);
```

```
return 0;
}
```

```
//Example 51
#include<stdio.h>
int main()
{
int x;
x=-2147483648;
printf("%d",x);
return 0;
}
```

```
//Example 52
#include<stdio.h>
int main()
{
int x;
x=-2147483649;
printf("%d",x);
return 0;
}
```

```
//Example53
#include<stdio.h>
int main()
{
char x;
x=128;
printf("%d",x);
return 0;
}
```

```
//Example 54
#include<stdio.h>
int main()
{
char x;
x=129;
printf("%d",x);
return 0;
}
```

```
//Example 55
#include<stdio.h>
int main()
{
char x;
x=-130;
printf("%d",x);
return 0;
```

```
}
```

```
//Example 56
#include<stdio.h>
int main()
{
char x;
x=-131;
printf("%d",x);
return 0;
}
```

```
//Example 57
#include<stdio.h>
int main()
{
char x;
x=256;
printf("%d",x);
return 0;
}
```

```
//Example 58
#include<stdio.h>
int main()
{
char x;
x=256;
printf("%d",x);
return 0;
}
```

```
//Example 59
#include<stdio.h>
int main()
{
char x;
x=257;
printf("%d",x);
return 0;
}
```

```
//Example 60
#include<stdio.h>
int main()
{
char x;
x=456;
printf("%d",x);
return 0;
}
```

```
//Example 61
#include<stdio.h>
int main()
{
char x;
x=512;
printf("%d",x);
return 0;
}
```

```
//Example 62
#include<stdio.h>
int main()
{
char x;
x=768;
printf("%d",x);
return 0;
}
```

```
//Example 63
#include<stdio.h>
int main()
{
char x;
x=767;
printf("%d",x);
return 0;
}
```

```
//Example 64
#include<stdio.h>
int main()
{
char x;
x=127;
x=x<<1;
printf("%d",x);
return 0;
}
```

```
//Example 65
#include<stdio.h>
int main()
{
char x;
x=127;
x=x<<2;
printf("%d",x);
return 0;
```

```
}
```

```
//Example 66
#include<stdio.h>
int main()
{
char x;
x=127;
x=x>>1;
printf("%d",x);
return 0;
}
```

```
//Example 67
#include<stdio.h>
int main()
{
char x;
x=127;
x=x>>2;
printf("%d",x);
return 0;
}
```

```
//Example 68
#include<stdio.h>
int main()
{
char x;
x=127;
x=x<<7;
printf("%d",x);
return 0;
}
```

```
//Example 69
#include<stdio.h>
int main()
{
char x;
x=1;
x=x<<2;
printf("%d",x);
return 0;
}
```

```
//Example 70
#include<stdio.h>
int main()
{
char x;
```

```
x=127;
x=x>>7;
printf("%d",x);
return 0;
}
```

```
//Example 71
#include<stdio.h>
int main()
{
char x,y,z,p,q;
x=128;
printf("%d\n",x);
y=128;
printf("%d\n",y);
p=-256;
printf("%d\n",p);
q=256;
printf("%d\n",q);
z=x+y;
printf("%d\n",z);
return 0;
}
```

```
//Example 72
#include<stdio.h>
int main()
{
char x,y,z;
x=511;
y=7;
z=x+y;
printf("%d",z);
return 0;
}
```

```
//Example 73
#include<stdio.h>
int main()
{
char z;
z=511+7;
printf("%d\n",z);
return 0;
}
```

```
//Example 74
#include<stdio.h>
int main()
{
```

```
int x;
x=64|32;
printf("%d",x);
return 0;
}
```

```
//Example 75
#include<stdio.h>
int main()
{
int x;
x=64&32;
printf("%d",x);
return 0;
}
```

```
//Example 76
#include<stdio.h>
int  main()
{
int x;
x=127&31;
printf("%d",x);
return 0;
}
```

```
//Example 77
#include<stdio.h>
int main()
{
int x;
x=65&40;
printf("%d",x);
return 0;
}
```

```
//Example 78
#include<stdio.h>
int main()
{
int x,y;
x=64;
y=65 & ++x;
printf("%d",y);
printf("%d\n",x);
return 0;
}
```

```
//Example 79
#include<stdio.h>
int main()
```

```
{
int x,y;
x=64;
y=65 & x++;
printf("%d",y);
return 0;
}
```

```
//Example 80
#include<stdio.h>
int main()
{
char x,y;
x=512;
y=x | 5;
printf("%d",y);
return 0;
}
```

```
//Example 81
#include<stdio.h>
int main()
{
char x,y;
x=512;
y=x & 5;
printf("%d",y);
return 0;
}
```

```
//Example 82
#include<stdio.h>
int main()
{
printf("%d\n",65 | 65);
printf("%d\n",65 | 32);
printf("%d\n",65 ^ 65);
printf("%d\n",65 ^ 32);
return 0;
}
```

```
//Example 83
#include<stdio.h>
int main()
{
int x;
x|=63 & 65;
printf("%d\n",x);
x=50;
x|=63 & 65;
```

```
printf("%d\n",x);
x&=73 | 75;
printf("%d\n",x);
return 0;
}
```

```
//Example 84
#include<stdio.h>
int main()
{
printf("%d %d\n",64,~64);
printf("%d %d\n",65,~65);
printf("%d %d\n",512,~512);
printf("%d %d\n",-64,~-64);
printf("%d %d\n",-65,~-65);
printf("%d %d\n",-512,~-512);
printf("%d %d\n",0,~0);
return 0;
}
```

```
//Example 85
#include<stdio.h>
int main()
{
char m[6];
printf("%d %d\n",sizeof(m),sizeof(m[0]));
printf("%d\n",m);
printf("%u\n",m);
m[0]='A';
m[1]='B';
m[3]='C';
m[4]='D';
m[5]='E';
printf("%d %d\n",sizeof(&m[0]),sizeof(m[0]));
printf("%d\n",&m[0]);
printf("%u\n",&m[0]);
return 0;
}
```

```
// Example 86
#include<stdio.h>
int main()
{
int x,y,count;
printf("Enter a number ");
scanf("%d",&x);
count=0;
for(y=x;y;y=y>>1) if(y & 1)count++;
printf("%d\n",count);
```

```
return 0;
}
```

```
//Example 87
#include<stdio.h>
#include<stdarg.h>
int add(int num,...)
{
va_list valist;
int sum=0;
int i;
va_start(valist,num); // num because after this parameter variable, … starts
for(i=0;i<num;i++)
{
sum+=va_arg(valist,int);
}
va_end(valist);
return sum;
}
int main()
{
printf("%d\n",add(3,10,20,30));
printf("%d\n",add(2,50,30));
printf("%d\n",add(7,10,20,30,40,50,60,70));
return 0;
}
```

```
// Example 88
#include<malloc.h>
#include<string.h>
#include<stdio.h>
#include<stdarg.h>
char * sam(char *string,...)
{
va_list valist;
int bufferSize=50;
char *arr=(char *)malloc(sizeof(char)*bufferSize);
char c,d;
char number[21];
int numberLength;
int i,ai;
va_start(valist,string);
i=0;
ai=0;
while(string[i]!='\0')
{
if(string[i]!='%')
{
```

```
arr[ai]=string[i];
ai++;
if(ai==bufferSize-1 && string[i+1]!='\0')
{
bufferSize+=50;
arr=(char *)realloc(arr,bufferSize);
}
i++;
continue;
}
if(string[i]=='%' && string[i+1]!='\0')
{
if(string[i+1]=='d')
{
itoa(va_arg(valist,int),number,10);
numberLength=strlen(number);
if(ai+numberLength>bufferSize-1)
{
bufferSize=bufferSize+50;
arr=(char *)realloc(arr,bufferSize);
}
strcpy(arr+ai,number);
ai+=numberLength;
i++;
}
else if(string[i+1]=='c')
{

}
else if(string[i+1]=='s')
{

}
else if(string[i+1]=='u')
{

}
else
{
arr[ai]=string[i];
ai++;
if(ai==bufferSize-1 && string[i+1]!='\0')
{
bufferSize+=50;
arr=(char *)realloc(arr,bufferSize);
}
}
```

```
}
i++;
}
va_end(valist);
arr[ai]='\0';
return arr;
}
int main()
{
char a[301];
char *p=sam("God %d is %d %rABgreat",10,20234);
strcpy(a,p);
printf("%s",a);
free(p);
return 0;
}
// Assignment complete the above example
```

```
// Example 89
#include<stdio.h>
int main()
{
int m[6];
printf("%d %d\n",sizeof(m),sizeof(m[0]));
printf("%d\n",m);
printf("%u\n",m);
printf("%u\n",&m);
m[0]=65;
m[1]=66;
m[3]=67;
m[4]=68;
m[5]=69;
printf("%d %d\n",sizeof(&m[0]),sizeof(m[0]));
printf("%d\n",&m[0]);
printf("%u\n",&m[0]);
return 0;
}
```

```
// Example 90
#include<stdio.h>
int main()
{
int m[6];
m[0]=65;
m[1]=66;
m[3]=67;
m[4]=68;
m[5]=69;
```

```
printf("%u\n",m);
printf("%u\n",&m[0]);
printf("%u\n",&m[4]);
printf("%u\n",m+4);
printf("%u\n",&m + 4);
return 0;
}
```

```
//Example 91
#include<stdio.h>
int main()
{
int m[3][6];
printf("%d\n",sizeof(m));
printf("%u\n",m);
printf("%u\n",m[0]);
printf("%u\n",&m[0]);
printf("%u\n",m+1);
printf("%u\n",m[1]);
printf("%u\n",&m[1]);
printf("%u\n",m+2);
printf("%u\n",m[2]);
printf("%u\n",&m[2]);
return 0;
}
```

```
// Example 92
#include<stdio.h>
int main()
{
char m[5];
printf("Size of m is %d\n",sizeof(m));
printf("Size of m[0] is %d\n",sizeof(m[0]));
printf("Address retrieved through &m is %u\n",&m);
printf("Address retrieved through m is %u\n",m);
printf("Address retrieved through &m[0] is %u\n",m);
printf("-------------------------\n");
printf("m+1 gives %u\n",m+1);
printf("&m[0]+1 gives %u\n",&m[0]+1);
printf("&m[1] gives %u\n",&m[1]);
printf("&m+1 gives %u\n",&m+1);
printf("-------------------------\n");
printf("m+2 gives %u\n",m+2);
printf("&m[0]+2 gives %u\n",&m[0]+2);
printf("&m[2] gives %u\n",&m[2]);
printf("&m+2 gives %u\n",&m+2);
printf("-------------------------\n");
printf("m+3 gives %u\n",m+3);
```

```
printf("&m[0]+3 gives %u\n",&m[0]+3);
printf("&m[3] gives %u\n",&m[3]);
printf("&m+3 gives %u\n",&m+3);
printf("-------------------------\n");
printf("m+4 gives %u\n",m+4);
printf("&m[0]+4 gives %u\n",&m[0]+4);
printf("&m[4] gives %u\n",&m[4]);
printf("&m+4 gives %u\n",&m+4);
printf("-------------------------\n");
printf("m+5 gives %u\n",m+5);
printf("&m[0]+5 gives %u\n",&m[0]+5);
printf("&m[5] gives %u\n",&m[5]);
printf("&m+5 gives %u\n",&m+5);
printf("-------------------------\n");
return 0;
}
```

```
// Example 93
#include<stdio.h>
int main()
{
int m[5];
printf("Size of m is %d\n",sizeof(m));
printf("Size of m[0] is %d\n",sizeof(m[0]));
printf("Address retrieved through &m is %u\n",&m);
printf("Address retrieved through m is %u\n",m);
printf("Address retrieved through &m[0] is %u\n",m);
printf("-------------------------\n");
printf("m+1 gives %u\n",m+1);
printf("&m[0]+1 gives %u\n",&m[0]+1);
printf("&m[1] gives %u\n",&m[1]);
printf("&m+1 gives %u\n",&m+1);
printf("-------------------------\n");
printf("m+2 gives %u\n",m+2);
printf("&m[0]+2 gives %u\n",&m[0]+2);
printf("&m[2] gives %u\n",&m[2]);
printf("&m+2 gives %u\n",&m+2);
printf("-------------------------\n");
printf("m+3 gives %u\n",m+3);
printf("&m[0]+3 gives %u\n",&m[0]+3);
printf("&m[3] gives %u\n",&m[3]);
printf("&m+3 gives %u\n",&m+3);
printf("-------------------------\n");
printf("m+4 gives %u\n",m+4);
printf("&m[0]+4 gives %u\n",&m[0]+4);
printf("&m[4] gives %u\n",&m[4]);
printf("&m+4 gives %u\n",&m+4);
```

```
printf("-------------------------\n");
printf("m+5 gives %u\n",m+5);
printf("&m[0]+5 gives %u\n",&m[0]+5);
printf("&m[5] gives %u\n",&m[5]);
printf("&m+5 gives %u\n",&m+5);
printf("-------------------------\n");
return 0;
}
```

```
//Example 94
#include<stdio.h>
struct abc
{
int x,y;
char g,t;
};
int main()
{
struct abc m[5];
printf("Size of m is %d\n",sizeof(m));
printf("Size of m[0] is %d\n",sizeof(m[0]));
printf("Address retrieved through &m is %u\n",&m);
printf("Address retrieved through m is %u\n",m);
printf("Address retrieved through &m[0] is %u\n",m);
printf("-------------------------\n");
printf("m+1 gives %u\n",m+1);
printf("&m[0]+1 gives %u\n",&m[0]+1);
printf("&m[1] gives %u\n",&m[1]);
printf("&m+1 gives %u\n",&m+1);
printf("-------------------------\n");
printf("m+2 gives %u\n",m+2);
printf("&m[0]+2 gives %u\n",&m[0]+2);
printf("&m[2] gives %u\n",&m[2]);
printf("&m+2 gives %u\n",&m+2);
printf("-------------------------\n");
printf("m+3 gives %u\n",m+3);
printf("&m[0]+3 gives %u\n",&m[0]+3);
printf("&m[3] gives %u\n",&m[3]);
printf("&m+3 gives %u\n",&m+3);
printf("-------------------------\n");
printf("m+4 gives %u\n",m+4);
printf("&m[0]+4 gives %u\n",&m[0]+4);
printf("&m[4] gives %u\n",&m[4]);
printf("&m+4 gives %u\n",&m+4);
printf("-------------------------\n");
printf("m+5 gives %u\n",m+5);
printf("&m[0]+5 gives %u\n",&m[0]+5);
```

```
printf("&m[5] gives %u\n",&m[5]);
printf("&m+5 gives %u\n",&m+5);
printf("-------------------------\n");
return 0;
}
```

```
// Example 95
#include<stdio.h>
int main()
{
int x[5];
int *p;
p=x;
p=&x[0];
p[0]=60;
printf("%d\n",x[0]);
*(p+4)=100;
printf("%d\n",x[4]);
*(x+2)=200;
return 0;
}
```

```
//Example 96
#include<stdio.h>
int  main()
{
int x[5];
int *p;
int **j;
p=x;
j=&x;
return 0;
}
```

```
//Example 97
#include<stdio.h>
int main()
{
int x[5];
int (*j)[5]; // type of j is int[5] *
j=&x;
printf("%u\n",&x);
printf("%u\n",j);
printf("%u\n",&x+1);
printf("%u\n",j+1);
return 0;
}
```

```
// Example 98
#include<stdio.h>
```

```
int main()
{
int x[5];
*(&x[0])=10;
printf("%d\n",x[0]);
return 0;
}
```

```
//Example 99
#include<stdio.h>
int main()
{
int x[5];
int *j=&x;
return 0;
}
```

```
//Example 100
#include<stdio.h>
int thinkingmachines(char *);
int main()
{
char a[81];
printf("Enter a string");
gets(a);
printf("%d\n",thinkingmachines(a));
return 0;
}
int thinkingmachines(char *p)
{
int k=0;
while(*p!='\0')
{
p++;
k++;
}
return k;
}
```

```
//Example 101
#include<stdio.h>
int thinkingmachines(char *);
int main()
{
char a[21];
printf("Enter a string");
gets(a);
printf("%d\n",thinkingmachines(a));
return 0;
```

```
}
int thinkingmachines(char *p)
{
char *q;
for(q=p;*q!='\0';q++);
return q-p;
}
```

```
// Example 102
#include<stdio.h>
void thinkingmachines(char *);
int main()
{
char a[81];
printf("Enter a string");
gets(a);
thinkingmachines(a);
printf("[%s]\n",a);
return 0;
}
void thinkingmachines(char *p)
{
char *q=p;
while(*q==' ')q++;
while(*q!='\0') {
*p=*q;
p++;
q++;
}
*p='\0';
}
```

```
// Example 103
#include<stdio.h>
void thinkingmachines(char *);
int main()
{
char a[81];
printf("Enter a string");
gets(a);
thinkingmachines(a);
printf("[%s]\n",a);
return 0;
}
void thinkingmachines(char *p)
{
char *q=p;
while(*q!='\0')
```

```
q++;
q--;
while(q>=p && *q==' ')
q--;
*(q+1)='\0';
}
```

```
//Example 104
#include<stdio.h>
void thinkingmachines(char *);
int main()
{
char a[81];
printf("Enter a string");
gets(a);
thinkingmachines(a);
printf("[%s]\n",a);
return 0;
}
void thinkingmachines(char *p)
{
char *q=p;
while(*q!='\0')
q++;
q--;
while(q>=p && *q==' ')
q--;
*(q+1)='\0';
q=p;
while(*q==' ')q++;
while(*q!='\0') {
*p=*q;
p++;
q++;
}
*p='\0';
}
```

```
// Example 105
#include<stdio.h>
void thinkingmachines(char *);
int main()
{
char a[81];
printf("Enter a string");
gets(a);
thinkingmachines(a);
printf("[%s]\n",a);
```

```
return 0;
}
void thinkingmachines(char *p)
{
char *q,*j,*m;
q=p;
while(*q!='\0')
{
for(j=m=q;*m==' ';m++);
if(j!=m){
if(j!=p)
j++;
while(*m!='\0'){
*j=*m;
j++;
m++;
}
*j='\0';
}
q++;
}
if(*(q-1)==' ')
*(q-1)='\0';
}
```

```
//Example 106
//Example 122
#include<stdio.h>
void thinkingmachines(char *,char *);
int main()
{
char a[81],b[81];
printf("Enter a string");
gets(a);
thinkingmachines(b,a);
printf("%s",b);
return 0;
}
void thinkingmachines(char *p,char *q)
{
while(*q!='\0')
{
*p=*q;
p++;
q++;
}
*p='\0';
```

```
}
```

```
// Example 107
#include<stdio.h>
void thinkingmachines(char *,char *);
int main()
{
char a[81],b[81];
printf("Enter a string");
gets(a);
thinkingmachines(b,a);
printf("%s",b);
return 0;
}
void thinkingmachines(char *p,char *q)
{
for(;*p=*q;p++,q++);
}
```

```
// Example 108
#include<stdio.h>
void thinkingmachines(char *,char *);
int main()
{
char a[81],b[81];
printf("Enter a string");
gets(a);
printf("Enter another string");
gets(b);
thinkingmachines(a,b);
printf("%s",a);
return 0;
}
void thinkingmachines(char *p,char *q)
{
while(*p!='\0')
p++;
while(*q!='\0'){
*p=*q;
p++;
q++;
}
*p='\0';
}
```

```
//Example 109
#include<stdio.h>
void thinkingmachines(char *,char *);
int main()
```

```
{
char a[81],b[81];
printf("Enter a string");
gets(a);
printf("Enter another string");
gets(b);
thinkingmachines(a,b);
printf("%s",a);
return 0;
}
void thinkingmachines(char *p,char *q)
{
while(*p!='\0') p++;
for(;*p=*q;p++,q++);
}
```

```
// Example 110
#include<stdio.h>
void thinkingmachines(char *);
int main()
{
char a[81];
printf("Enter a string");
gets(a);
thinkingmachines(a);
printf("%s",a);
return 0;
}
void thinkingmachines(char *p)
{
char *q;
for(q=p;*(q+1);q++);
while(p<q){
*p=*p+*q;
*q=*p-*q;
*p=*p-*q;
p++;
q--;
}
}
```

```
// Example 111
#include<stdio.h>
void thinkingmachines(char *);
int main()
{
char a[81];
printf("Enter a string");
```

```
gets(a);
thinkingmachines(a);
printf("%s",a);
return 0;
}
void thinkingmachines(char *p)
{
int x;
for(x=0;*(p+x+1);x++);
while(p<p+x)
{
*p=*p+*(p+x);
*(p+x)=*p-*(p+x);
*p=*p-*(p+x);
x-=2;
p++;
}
}
```

```
// Example 112
#include<stdio.h>
void thinkingmachines(char *);
int main()
{
char a[81];
printf("Enter a string");
gets(a);
thinkingmachines(a);
printf("%s",a);
return 0;
}
void thinkingmachines(char *p)
{
char *q=p;
while(*q)
{
*q=((p==q || (q>p && *(q-1)==' ')) && *q>=97 && *q<=122)?*q-=' ':*q;
q++;
}
}
```

```
// Example 113
#include<stdio.h>
int thinkingmachines(char *,char *);
int main()
{
char a[81],b[11];
int x;
```

```
printf("Enter a string");
gets(a);
printf("Enter another string");
gets(b);
x=thinkingmachines(a,b);
printf("%d",x);
return 0;
}
int thinkingmachines(char *p,char *q)
{
int x=0;
char *j1,*j2;
while(*p!='\0')
{
for(j1=p,j2=q;*j1==*j2 && *j1 && *j2;j1++,j2++);
if(!*j2)
x++;
p++;
}
return x;
}
```

```
//Example 114
#include<stdio.h>
void thinkingmachines(char *,char *);
int main()
{
char a[81],b[11];
int x;
printf("Enter a string");
gets(a);
printf("Enter another string");
gets(b);
thinkingmachines(a,b);
printf("%s",a);
return 0;
}
void thinkingmachines(char *p,char *q)
{
char *j1,*j2;
while(*p!='\0')
{
for(j1=p,j2=q;*j1==*j2 && *j1 && *j2;j1++,j2++);
if(!*j2)
{
for(j1=p,j2=p+(j2-q);*j2;j1++,j2++) {
*j1=*j2;
```

```
}
*j1='\0';
if(p==j1)
break;
}
p++;
}
}
```

```
// Example 115
#include<stdio.h>
void lmn();
int main()
{
lmn();
lmn();
lmn();
}
void lmn()
{
int x=1;
printf("%d\n",x);
x++;
}
```

```
//Example 116
#include<stdio.h>
void lmn();
int main()
{
lmn();
lmn();
lmn();
}
void lmn()
{
static int x;
x=1;
printf("%d\n",x);
x++;
}
```

```
//Example 117
#include<stdio.h>
void lmn();
int main()
{
lmn();
lmn();
```

```
lmn();
}
void lmn()
{
static int x=1;
printf("%d\n",x);
x++;
}
```

//Example 118

anil.h

```
void cool();
```

anil.c

```
#include<stdio.h>
int x;
void cool()
{
printf("Cool");
}
```

To compile and create library file

gcc -c anil.c

ar rcs libanil.lib anil.o

eg118.c

```
#include "anil.h"
int main()
{
x=10;
cool();
return 0;
}
```

Now compile the above example

gcc -static eg118.c -l libanil -L . -o eg118.exe

The code won't compile

Now change the code

```
#include "anil.h"
extern int x;
int main()
{
x=10;
cool();
return 0;
}
```

gcc -static eg118.c -l libanil -L . -o eg118.exe

This time the code will compile.

```
//Example 119
#include<stdio.h>
int main()
{
int x[10];
int lookFor,found,y;
y=0;
while(y<=9)
{
printf("Enter a number");
scanf("%d",&x[y]);
y++;
}
printf("Enter the number to look for ");
scanf("%d",&lookFor);
y=0;
found=0;
while(y<=9)
{
if(x[y]==lookFor)
{
found=1;
break;
}
y++;
}
if(found==0)
{
printf("Not found");
}
else
{
printf("Found");
}

return 0;
}
```

```
//Example 120
#include<stdio.h>
int main()
{
int x[10];
int lookFor,found,y;
y=0;
while(y<=9)
{
```

```
printf("Enter a number");
scanf("%d",&x[y]);
y++;
}
printf("Enter the number to look for ");
scanf("%d",&lookFor);
y=0;
found=0;
while(y<=9)
{
if(x[y]==lookFor)
{
found=1;
break;
}
y++;
}
if(found==0)
{
printf("Not found");
}
else
{
printf("Found at index %d",y);
}
return 0;
}
```

```
//Example 121
#include<stdio.h>
int main()
{
int x[10],y,lookFor,count;
y=0;
while(y<=9)
{
printf("Enter a number");
scanf("%d",&x[y]);
y++;
}
printf("Enter the number to look for ");
scanf("%d",&lookFor);
y=0;
count=0;
while(y<=9)
{
if(lookFor==x[y])
```

```
{
count++;
}
y++;
}
if(count==1)
{
printf("%d occurs 1 time",lookFor);
}
else
{
printf("%d occurs %d times",lookFor,count);
}
return 0;
}
```

```
//Example 122
#include<stdio.h>
int main()
{
int x[10],y,lookFor,count;
y=0;
while(y<=9)
{
printf("Enter a number");
scanf("%d",&x[y]);
y++;
}
lookFor=x[0];
y=0;
count=0;
while(y<=9)
{
if(lookFor==x[y])
{
count++;
}
y++;
}
if(count==1)
{
printf("%d occurs 1 time",lookFor);
}
else
{
printf("%d occurs %d times",lookFor,count);
}
```

```
return 0;
}
```

```
//Example 123
#include<stdio.h>
int main()
{
int x[10],y,lookFor,count,z;
y=0;
while(y<=9)
{
printf("Enter a number");
scanf("%d",&x[y]);
y++;
}
z=0;
while(z<=9)
{
lookFor=x[z];
y=0;
count=0;
while(y<=9)
{
if(lookFor==x[y])
{
count++;
}
y++;
}
if(count==1)
{
printf("%d occurs 1 time\n",lookFor);
}
else
{
printf("%d occurs %d times\n",lookFor,count);
}
z++;
}
return 0;
}
```

```
//Example 124
#include<stdio.h>
int main()
{
int x[10],y,lookFor,count,z,found,e;
y=0;
```

```
while(y<=9)
{
printf("Enter a number");
scanf("%d",&x[y]);
y++;
}
z=0;
while(z<=9)
{
lookFor=x[z];
found=0;
e=0;
while(e<=z-1)
{
if(lookFor==x[e])
{
found=1;
break;
}
e++;
}
if(found==0)
{
count=1;
y=z+1;
while(y<=9)
{
if(lookFor==x[y])
{
count++;
}
y++;
}
if(count==1)
{
printf("%d occurs 1 time\n",lookFor);
}
else
{
printf("%d occurs %d times\n",lookFor,count);
}
}
z++;
}
return 0;
}
```

```
//Example 125
#include<stdio.h>
typedef  char name[21];
typedef  char address[101];
int main()
{
name s; // char s[21];
address g; // char g[101];
return 0;
}
```

```
//Example 126
#include<stdio.h>
int main()
{
int x;
x=10;
x=20;
printf("%d\n",x);
return 0;
}
```

```
//Example 127
#include<stdio.h>
int main()
{
const int x;
x=10;
x=20;
printf("%d\n",x);
return 0;
}
```

```
//Example 128
#include<stdio.h>
int main()
{
const int x=5;
printf("%d\n",x);
return 0;
}
```

```
//Example 129
#include<stdio.h>
int main()
{
int x,y;
int *p;
p=&x;
*p=50;
```

```
p=&y;
*p=30;
printf("%d\n",x);
printf("%d\n",y);
return 0;
}
```

```
//Example 130
#include<stdio.h>
int main()
{
int x,y;
int *p;
p=&x;
*p=50;
p=&y;
*p=30;
printf("%d\n",x);
printf("%d\n",y);
return 0;
}
```

```
//Example 131
#include<stdio.h>
int main()
{
int x,y;
const int *p=&x;
p=&y;
*p=60;
return 0;
}
```

```
//Example 132
#include<stdio.h>
int main()
{
int const y=3;
return 0;
}
```

```
//Example 133
#include<Stdio.h>
#define uj printf
int main()
{
uj("hello");
return 0;
}
```

```
//Example 134
```

```
#include<Stdio.h>
#define uj printf
int main()
{
uj("ujjain is a cool city");
return 0;
}
```

```
//Example 135
#include<Stdio.h>
#define pr printf
int main()
{
pr("ujjain is a cool city");
printf("God is great");
return 0;
}
```

```
//Example 136
#include<Stdio.h>
#define x printf
int main()
{
int x;
x("god is great");
return 0;
}
```

```
//Example 137
#include<Stdio.h>
int main()
{
int printf;
printf("god is great");
return 0;
}
```

```
//Example 138
#include<Stdio.h>
int main()
{
int k;
k("god is great");
return 0;
}
```

```
//Example 139
#include<Stdio.h>
#define max(x,y) (x>y)?x:y
int main()
{
```

```
int j,t;
j=100;
t=200;
printf("%d",max(j,t));
return 0;
}
```

```
//Example 140
#include<Stdio.h>
int main()
{
char m;
m='A';
if((m>=48 && m<=57)?1:0){
printf("%c is a digit\n",m);
} else {
printf("%c is not a digit\n",m);
}
return 0;
}
```

```
//Example 141
#include<Stdio.h>
#define isdigit(x) (x>=48 && x<=57)?1:0
#define isnondigit(x) !isdigit(x)
int main()
{
char m;
m='A';
if(isdigit(m)){
printf("%c is a digit\n",m);
} else {
printf("%c is not a digit\n",m);
}
if(isnondigit(m)){
printf("%c is not a digit\n",m);
} else {
printf("%c is  a digit\n",m);
}
return 0;
}
```

```
// Example 142
#include<Stdio.h>
#define TRUE 1
#define FALSE 0
#define isdigit(x) (x>=48 && x<=57)?TRUE:FALSE
#define isnondigit(x) !isdigit(x)
int main()
```

```
{
char m;
m='A';
if(isdigit(m)){
printf("%c is a digit\n",m);
} else {
printf("%c is not a digit\n",m);
}
if(isnondigit(m)){
printf("%c is not a digit\n",m);
} else {
printf("%c is  a digit\n",m);
}
return 0;
}
```

```
// Example 143
#include<Stdio.h>
#define TRUE 1
#define FALSE 0
#define isdigit(x) (x>=48 && x<=57)?TRUE:FALSE
#define isnondigit(x) !isdigit(x)
int main()
{
char m;
m='A';
if(isdigit(m)){
printf("%c is a digit\n",m);
} else {
printf("%c is not a digit\n",m);
}
if(isnondigit(m)){
printf("%c is not a digit\n",m);
} else {
printf("%c is  a digit\n",m);
}
return 0;
}
```

```
//Example 144
#include<stdio.h>
#include<alloc.h>
static int p; // will get default value -> 0 storage -> data segment
int j;
extern int r; // memory won't be allocated for r
void lmn();
int main()
{
```

```
register int m; // will get space in CPU memory
lmn();
return 0;
}
void lmn()
{
int *g; // stack - remainingmemory
static int x; // will get default value -> 0 -> data segment
int y; // stack - remianing memory
auto int z; // same as y // stack - remaining memory
printf("%d\n",p);
printf("%d\n",x);
g=(int *)malloc(2); // allocation in heap (remaining / data segment)
g=(int *)calloc(10,2); // allocation in heap (remaining / data segment)
}
// when a program is executed the functions are loaded in memory
// and all the instructions are stored in code segment
```

```
//Example 145
#include<stdio.h>
#include<String.h>
int main()
{
char a[81],b[81];
char *p;
int x;
strcpy(a,"Ujjain");
strcpy(b,"Goa");
printf("%s\n",b);
strcat(a,b);
printf("%s\n",a);
p=strchr(a,'j');
if(p==NULL) {
printf("Not found");
}
else {
printf("Found at index %d\n",p-a);
printf("%s\n",p);
}
p=strchr(a,'z');
if(p==NULL) {
printf("Not found\n");
}
else {
printf("Found at index %d\n",p-a);
printf("%s\n",p);
}
```

```
p=strrchr(a,'j');
if(p==NULL) {
printf("Not found");
}
else {
printf("Found at index %d\n",p-a);
printf("%s\n",p);
}
strcpy(a,"Mumbai");
strcpy(b,"mumbai");
x=strcmp(a,b); // case sensitive comparison
printf("%d\n",x); // 0, greater than 0 or less than 0
x=strcmpi(a,b); // incase sensitive comparison
printf("%d\n",x); // 0, greater than 0 or less than 0
printf("%d\n",strlen(a));
p=strdup(a);
printf("%s\n",p);
strlwr(a);
printf("%s\n",a);
strupr(a);
printf("%s\n",a);
strncat(a,b,3);
printf("%s\n",a);
strcpy(a,"bombay");
strcpy(b,"bombarding");
x=strncmp(a,b,4);
x=strnicmp(a,b,4);
x=strncmpi(a,b,4);
strcpy(a,"Ujjain");
strcpy(b,"goa");
printf("%s\n",b);
strncpy(b,a,3);
printf("%s\n",b);
strncpy(b,a,5);
printf("%s\n",b);
strcpy(a,"Mumbai");
strset(a,'z');
printf("%s\n",a);
strcpy(a,"Mumbai");
strnset(a,'z',3);
printf("%s\n",a);
strcpy(a,"Mumbai");
strrev(a);
printf("%s\n",a);
strcpy(a,"mumbaitogoa");
p=strstr(a,"to");
printf("%s\n",p);
```

```
return 0;
}
```

```
//Example 146
#include<stdio.h>
#include<string.h>
int main()
{
char a[21];
printf("Enter a string");
scanf("%s",a);
if(strcmp(strrev(strdup(a)),a)==0) {
printf("%s","Pallindrome");
}
else {
printf("%s","Not a Pallindrome");
}
return 0;
}
```

List of programs that you should be able to write within minutes.

Factorial etc. (all examples that we learned while learning C)
Total of 10 numbers, find largest etc. (all examples that we learned while learning C)
Creating pyramids
Pascal's Triangle
Count number of on bits in a number
Accept a number and print if it is even or odd (you cannot use loop or if/else
Print system date and time
All implementations of Example 145 without using predefined functions
Reverse a number
Armstrong number

Data Structures

Bubble Sort
Selection Sort
Insertion Sort
Merge Sort
Quick Sort
Heap Sort
Radix Sort
Singly Linked List
operation : add a node at end
insert a node at top
insert a node at a particular position
remove a node from a particular position
traverse list (top to bottom)
traverse list (bottom to top) (using recursion and stack)

Doubly linked list (Same as above)
Binary Search Tree
operation : Add a node
traverse (Pre/Post/In Order) (using recursion and stack)
remove a node (12 cases)
AVL Tree (BST Operations + Balancing factor)
Stack implementation using Array
Stack implementation using Singly / Doubly Linked List
Queue implementation using Array
Queue implementation using Singly / Double Linked List
converting infix to postfix
evaluating a postfix expression

File Handling

In all following examples accept file name(s) as command line argument.

1) Count the number of bytes in a file
2) Find one string and replace it with another (all occurrences)
3) Merge two files into a third one.
4) Print number of characters, words and lines.
5) Print the words with their frequency (count).
6) Reverse all the words in the file.
7) Reverse the contents of the file (First word should become last word)
8) Indent the contents of a source code file.
9) Print distinct lines. (If a line is being repeated, it should be printed only once)
10) Read all lines into an array
11) Compare contents of 2 files to determine if they are same or not
12) Accept 3 file names (for eg. a,b and c) find the contents of b in a and replace it with c and print the number of occurrences replaced.
13) Accept the name of file that contains math expressions, process it and print the result of all the expressions one by one.

First of all download cppbookone.pdf and complete it
Then

Settings.h

```
#define boolean int
#define true 1
#define TRUE 1
#define false 0
#define FALSE 0
```

Student.h

```
#include<string.h>
class Student
{
private:
int rollNumber;
char *name;
public:
Student();
Student(const Student &);
Student & operator=(Student);
~Student();
void setRollNumber(int);
int getRollNumber();
void setName(const char *);
void getName(char * &);
};
Student::Student()
{
cout<<"Default constructor got invoked"<<endl;
this->rollNumber=0;
this->name=NULL;
}
Student::Student(const Student &otherStudent)
{
cout<<"Copy constructor got invoked"<<endl;
this->rollNumber=0;
this->name=NULL;
this->setRollNumber(otherStudent.rollNumber);
this->setName(otherStudent.name);
}
Student & Student::operator=(Student otherStudent)
{
cout<<"= function got invoked"<<endl;
this->setRollNumber(otherStudent.rollNumber);
this->setName(otherStudent.name);
return *this;
```

```
}
Student::~Student()
{
if(this->name!=NULL) delete [] this->name;
}
void Student::setRollNumber(int rollNumber)
{
this->rollNumber=rollNumber;
}
int Student::getRollNumber()
{
return this->rollNumber;
}
void Student::setName(const char *name)
{
if(this->name!=NULL)
{
delete [] this->name;
this->name=NULL;
}
if(name==NULL) return;
this->name=new char[strlen(name)+1];
strcpy(this->name,name);
}
void Student::getName(char * &name)
{
if(this->name==NULL)
{
name=NULL;
return;
}
name=new char[strlen(this->name)+1];
strcpy(name,this->name);
}
```

Employee.h

```
#include<string.h>
class Employee
{
private:
int employeeID;
char *name;
int age;
public:
Employee();
Employee(const Employee &);
Employee & operator=(Employee);
```

```
~Employee();
void setEmployeeID(int);
int getEmployeeID();
void setName(const char *);
void getName(char * &);
void setAge(int);
int getAge();
};
Employee::Employee()
{
this->employeeID=0;
this->name=NULL;
this->age=0;
}
Employee::Employee(const Employee &otherEmployee)
{
this->employeeID=0;
this->name=NULL;
this->age=0;
this->setEmployeeID(otherEmployee.employeeID);
this->setName(otherEmployee.name);
this->setAge(otherEmployee.age);
}
Employee & Employee::operator=(Employee otherEmployee)
{
this->setEmployeeID(otherEmployee.employeeID);
this->setName(otherEmployee.name);
this->setAge(otherEmployee.age);
return *this;
}
Employee::~Employee()
{
if(this->name!=NULL) delete [] this->name;
}
void Employee::setEmployeeID(int employeeID)
{
this->employeeID=employeeID;
}
int Employee::getEmployeeID()
{
return this->employeeID;
}
void Employee::setName(const char *name)
{
if(this->name!=NULL)
{
delete [] this->name;
```

```
this->name=NULL;
}
if(name==NULL) return;
this->name=new char[strlen(name)+1];
strcpy(this->name,name);
}
void Employee::getName(char * &name)
{
if(this->name==NULL)
{
name=NULL;
return;
}
name=new char[strlen(this->name)+1];
strcpy(name,this->name);
}
void Employee::setAge(int age)
{
this->age=age;
}
int Employee::getAge()
{
return this->age;
}
```

Singly.h

```
template<class T>
class SinglyLinkedListNode;
template<class T>
class SinglyLinkedListIterator;
template<class T>
class SinglyLinkedList;
template<class T>
class SinglyLinkedListNode
{
private:
T data;
SinglyLinkedListNode *next;
SinglyLinkedListNode(T);
friend class SinglyLinkedList<T>;
friend class SinglyLinkedListIterator<T>;
};
template<class T>
SinglyLinkedListNode<T>::SinglyLinkedListNode(T data)
{
this->data=data;
this->next=NULL;
```

```
}
template<class T>
class SinglyLinkedListIterator
{
private:
SinglyLinkedListNode<T> *node;
SinglyLinkedListIterator(SinglyLinkedListNode<T> *);
public:
boolean hasNext();
T next();
friend class SinglyLinkedList<T>;
};
template<class T>
SinglyLinkedListIterator<T>::SinglyLinkedListIterator(SinglyLinkedListNode<T> *node)
{
this->node=node;
}
template<class T>
boolean SinglyLinkedListIterator<T>::hasNext()
{
return this->node!=NULL;
}
template<class T>
T SinglyLinkedListIterator<T>::next()
{
if(this->node==NULL) return NULL;
T data;
data=this->node->data;
this->node=this->node->next;
return data;
}
template<class T>
class SinglyLinkedList
{
private:
SinglyLinkedListNode<T> *start;
SinglyLinkedListNode<T> *end;
int size;
public:
SinglyLinkedList();
SinglyLinkedList(const SinglyLinkedList<T> &);
~SinglyLinkedList();
SinglyLinkedList<T> & operator=(SinglyLinkedList<T>);
void add(T);
void insert(int,T);
T remove(int);
void clear();
```

```
T get(int);
int getSize();
SinglyLinkedListIterator<T> * getIterator();
};
template<class T>
SinglyLinkedList<T>::SinglyLinkedList()
{
this->size=0;
this->start=NULL;
this->end=NULL;
}
template<class T>
SinglyLinkedList<T>::SinglyLinkedList(const SinglyLinkedList<T> &otherSinglyLinkedList)
{
this->size=0;
this->start=NULL;
this->end=NULL;
SinglyLinkedListNode<T> *node;
node=otherSinglyLinkedList.start;
while(node!=NULL)
{
this->add(node->data);
node=node->next;
}
}
template<class T>
SinglyLinkedList<T>::~SinglyLinkedList()
{
this->clear();
}
template<class T>
SinglyLinkedList<T> & SinglyLinkedList<T>::operator=(SinglyLinkedList<T>
otherSinglyLinkedList)
{
this->clear();
SinglyLinkedListNode<T> *node;
node=otherSinglyLinkedList.start;
while(node!=NULL)
{
this->add(node->data);
node=node->next;
}
return *this;
}
template<class T>
void SinglyLinkedList<T>::add(T data)
{
```

```
SinglyLinkedListNode<T> *node;
node=new SinglyLinkedListNode<T>(data);
if(this->start==NULL)
{
this->start=node;
this->end=node;
}
else
{
end->next=node;
end=node;
}
this->size++;
}
template<class T>
T SinglyLinkedList<T>::get(int index)
{
if(index<0 || index>=size) return NULL;
SinglyLinkedListNode<T> *node;
node=this->start;
int x;
x=0;
while(x<index)
{
node=node->next;
x++;
}
return node->data;
}
template<class T>
int SinglyLinkedList<T>::getSize()
{
return this->size;
}
template<class T>
void SinglyLinkedList<T>::insert(int position,T data)
{
if(position<0) position=0;
if(position>this->size) position=this->size;
if(position==this->size)
{
this->add(data);
return;
}
SinglyLinkedListNode<T> *node;
node=new SinglyLinkedListNode<T>(data);
if(position==0)
```

```
{
node->next=this->start;
this->start=node;
}
else
{
SinglyLinkedListNode<T> *t,*k;
int x;
t=this->start;
x=0;
while(x<position)
{
k=t;
t=t->next;
x++;
}
node->next=t;
k->next=node;
}
this->size++;
}
template<class T>
T SinglyLinkedList<T>::remove(int position)
{
if(position<0 || position>=this->size) return NULL;
T data;
SinglyLinkedListNode<T> *t;
if(this->size==1)
{
t=this->start;
data=t->data;
this->start=NULL;
this->end=NULL;
delete t;
this->size=0;
return data;
}
SinglyLinkedListNode<T> *k;
t=this->start;
int x;
x=0;
while(x<position)
{
k=t;
t=t->next;
x++;
}
```

```
if(this->start==t)
{
this->start=this->start->next;
data=t->data;
delete t;
this->size--;
return data;
}
if(this->end==t)
{
end=k;
end->next=NULL;
data=t->data;
delete t;
this->size--;
return data;
}
k->next=t->next;
data=t->data;
delete t;
this->size--;
return data;
}
template<class T>
void SinglyLinkedList<T>::clear()
{
while(this->size>0) this->remove(0);
}
template<class T>
SinglyLinkedListIterator<T> * SinglyLinkedList<T>::getIterator()
{
return new SinglyLinkedListIterator<T>(this->start);
}
```

Now try creating templates to wrap other data structures like DoublyLinkedList, BinarySearchTree etc.