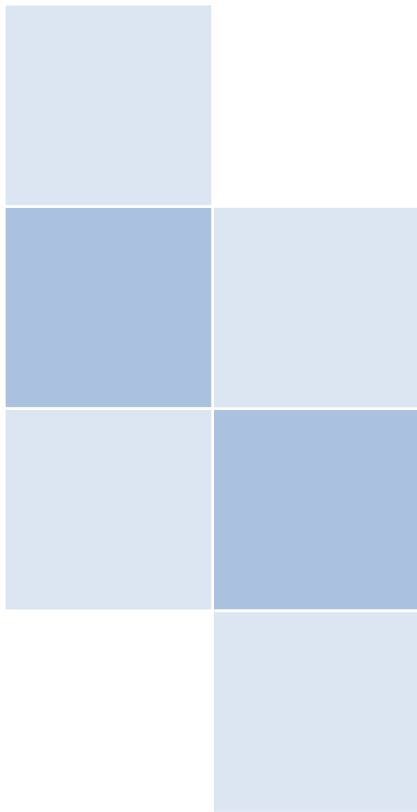


DATA STRUCTURES WITH C/C++ LAB MANUAL



Akash Devaraju

2011

[DEPARTMENT OF CS/IS]

Treating Code As An Essay

PROGRAMS SHARE SOME ATTRIBUTES WITH ESSAYS. For essays, the most important question readers ask is, “What is it about?” For programs, the main question is, “What does it do?” In fact, the purpose should be sufficiently clear that neither question ever needs to be uttered. Still, for both essays and computer code, it’s always important to look at how each one is written. Even if the idea itself is good, it will be difficult to transmit to the desired audience if it is difficult to understand. The style in which they are written is just as important as their purpose. Both essays and lines of code are meant—before all else—to be read and understood by human beings.

Computers can, of course, deal with complexity without complaint, but this is not the case for human beings. Unreadable code will reduce most people’s productivity significantly. On the other hand, easily understandable code will increase it. And we see beauty in such code.

“Beautiful Code” is not an abstract virtue that exists independent of its programmer’s efforts. Rather, beautiful code is really meant to help the programmer be happy & productive.

In other words, what is beautiful code?

Brevity: “Succinctness is power”, there’s a definite cost involved in scanning code with the human eye, and programs should ideally contain no unnecessary information. Eliminate Redundancy ...

DRY Principle: Don’t Repeat Yourself. If the same code exists in multiple places whatever you are trying to say becomes obscured ...

Simplicity: Feel beauty in simple code. If a program is hard to understand, it can’t be beautiful. And when programs are obscure rather than comprehensible, the results are bugs, mistakes & confusion.

Flexibility: “Freedom from enforcement of tools”. Humans are more valuable than any tools or languages, computers should serve programmers to maximise their productivity & happiness but in reality they often increase the burden instead of lightening it.

Balance is the final element of beautiful code. None of the above elements will by itself ensure a beautiful program. When balanced together & kept in mind from the very beginning, each element will work harmoniously with the others to create a beautiful code. And if you make sure to have fun writing and reading code, you will experience happiness as a programmer

Yukihiro (‘Matz’) Matsumoto

(Creator of the “Ruby” Language)

(Adapted from the book – “The Beautiful Code”)

Contents

TREATING CODE AS AN ESSAY	1
CONTENTS.....	2
LAB PROGRAM 1.....	4
OUTPUT.....	7
LAB PROGRAM 2.....	8
OUTPUT.....	10
LAB PROGRAM 3.....	11
OUTPUT.....	13
LAB PROGRAM 4.....	14
OUTPUT.....	16
LAB PROGRAM 5.....	18
OUTPUT.....	20
LAB PROGRAM 6.....	21
OUTPUT.....	23
LAB PROGRAM 7.....	24
OUTPUT.....	26
LAB PROGRAM 8.....	28
OUTPUT.....	30
LAB PROGRAM 9.....	32
OUTPUT.....	33
LAB PROGRAM 10.....	34
OUTPUT.....	36
LAB PROGRAM 11.....	38
OUTPUT.....	42
LAB PROGRAM 12.....	46
OUTPUT.....	49
LAB PROGRAM 13.....	50
OUTPUT.....	52
LAB PROGRAM 14.....	53
OUTPUT.....	56

Lab Program 1

```
1  /**
2   Lab Program 1
3   Circular Polynomial Addition
4   */
5
6   #include <stdio.h>
7   #include <stdlib.h>
8   #include <conio.h>
9
10  typedef struct Node
11  {
12      int exp;
13      int coef;
14      struct Node* link;
15  }Node;
16
17  typedef struct
18  {
19      Node* first;
20      Node* last;
21  }Polynomial;
22
23  Polynomial Insert(Polynomial P, int exp, int coef);
24  Polynomial Input(Polynomial P);
25  void Print(Polynomial P);
26  Polynomial Add(Polynomial A, Polynomial B, Polynomial C);
27
28  void main()
29  {
30      Polynomial a, b, c;
31      a.first = b.first = c.first = NULL;
32
33      a = Input(a);
34      b = Input(b);
35
36      printf("\n\nPolynomial A : ");
37      Print(a);
38
39      printf("\n\nPolynomial B : ");
40      Print(b);
41
42      c = Add(a, b, c);
43
44      printf("\n\nAdding A and B, Polynomial C : ");
45      Print(c);
46
47      getch();
48  }
```

```

51 Polynomial Insert(Polynomial P, int exp, int coef)
52 {
53     if(P.first==NULL)
54     {
55         P.first = P.last =(Node *)malloc(sizeof(Node));
56         P.last->link = P.first;
57     }
58     else
59     {
60         P.last->link =(Node *)malloc(sizeof(Node));
61         P.last = P.last->link;
62         P.last->link = P.first;
63     }
64
65     P.last->exp = exp;
66     P.last->coef = coef;
67
68     return P;
69 }
70
71 Polynomial Input(Polynomial P)
72 {
73     int degree, coef, i;
74     printf("\nEnter the degree of the polynomial: ");
75     scanf("%d", &degree);
76
77     printf("\n");
78     for(i = degree; i >= 0; i--)
79     {
80         printf("Enter the co-efficient of x^%d: ", i);
81         scanf("%d", &coef);
82         P = Insert(P, i, coef);
83     }
84
85     return P;
86 }
87
88 void Print(Polynomial P)
89 {
90     Node* temp = P.first;
91     do
92     {
93         if(temp->coef == 0)
94         {
95             temp = temp->link;
96             continue;
97         }
98
99         printf("%dx^%d ", temp->coef, temp->exp);
100         temp = temp->link;
101     }while(temp != P.first);
102
103     printf("\n");
104 }

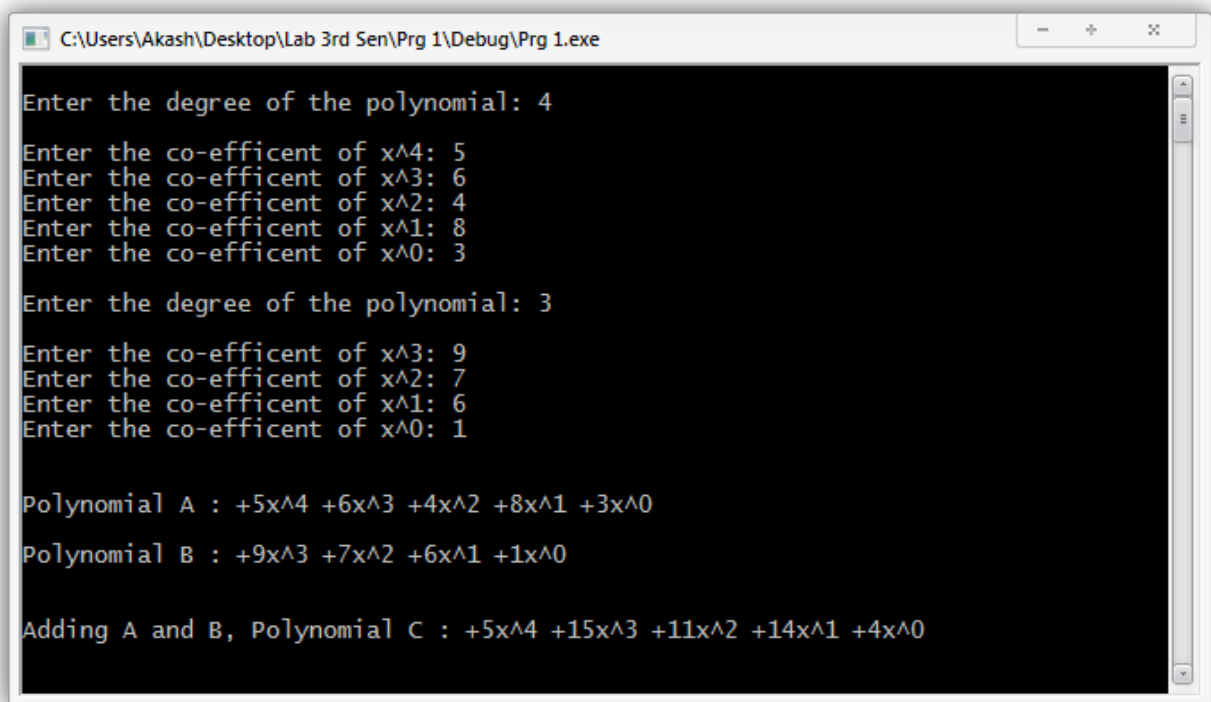
```

```

106 Polynomial Add(Polynomial A, Polynomial B, Polynomial C)
107 {
108     Node* tempA = A.first;
109     Node* tempB = B.first;
110     do
111     {
112         if(tempA->exp > tempB->exp)
113         {
114             C = Insert(C, tempA->exp, tempA->coef);
115             tempA = tempA->link;
116         }
117
118         else if(tempA->exp == tempB->exp)
119         {
120             C = Insert(C, tempA->exp, tempA->coef + tempB->coef);
121             tempA = tempA->link;
122             tempB = tempB->link;
123         }
124
125         else
126         {
127             C = Insert(C, tempB->exp, tempB->coef);
128             tempB = tempB->link;
129         }
130
131     }while(tempA != A.first || tempB != B.first);
132
133     return C;
134 }

```

Output



A screenshot of a Windows command prompt window titled "C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 1\Debug\Prg 1.exe". The window has a black background with white text. It shows the following sequence of inputs and outputs:

```
Enter the degree of the polynomial: 4
Enter the co-efficient of x^4: 5
Enter the co-efficient of x^3: 6
Enter the co-efficient of x^2: 4
Enter the co-efficient of x^1: 8
Enter the co-efficient of x^0: 3

Enter the degree of the polynomial: 3
Enter the co-efficient of x^3: 9
Enter the co-efficient of x^2: 7
Enter the co-efficient of x^1: 6
Enter the co-efficient of x^0: 1

Polynomial A : +5x^4 +6x^3 +4x^2 +8x^1 +3x^0
Polynomial B : +9x^3 +7x^2 +6x^1 +1x^0

Adding A and B, Polynomial C : +5x^4 +15x^3 +11x^2 +14x^1 +4x^0
```


Lab Program 2

```
1  /**
2   Lab Program 2
3   Infix To Postfix Expression
4   Author SkyK0G
5   */
6
7   #include <stdio.h>
8   #include <conio.h>
9
10  int stack[10];
11  int top=-1;
12
13  void push(char);
14  char pop();
15  int stpri(char ch);
16  int inpri(char ch);
17  void convert(char [],char[]);
18
19  void main()
20  {
21      char infix[20],postfix[20];
22      push('#');
23      printf("Enter Infix Expression : ");
24      fflush(stdin);
25      gets(infix);
26      convert(infix,postfix);
27      printf("\nThe Evaluated Postfix Expression is : %s",postfix);
28      getch();
29  }
30
31  void push(char ch)
32  {
33      stack[++top]=ch;
34  }
35
36  char pop()
37  {
38      return (stack[top--]);
39  }
40
41  int stpri(char ch)
42  {
43      switch(ch)
44      {
45          case '#':
46          case '(':return 0;
47          case '+':
48          case '-':return 2;
49          case '*':
50          case '/':return 4;
51          case '$':return 5;
52      }
53  }
```

```

55 int inpri(char ch)
56 {
57     switch(ch)
58     {
59         case '+':
60         case '-':return 1;
61         case '*':
62         case '/':return 3;
63         case '$':return 6;
64     }
65 }
66
67 void convert(char inf[20],char post[20])
68 {
69     int i,j=0;
70     char ch;
71     for(i=0;inf[i];i++)
72     {
73         ch=inf[i];
74         switch(ch)
75         {
76             case '(':push(ch);
77                 break;
78             case ')':while(stack[top]!='(')
79                 post[j++]=pop();
80                 pop();
81                 break;
82             case '+':
83             case '-':
84             case '/':
85             case '*':
86             case '$':while(stpri(stack[top])>inpri(ch))
87                 post[j++]=pop();
88                 push(ch);
89                 break;
90             default:post[j++]=ch;
91         }
92     }
93     while(stack[top]!='#')
94         post[j++]=pop();
95     post[j]='\0';
96 }

```

Output



A screenshot of a Windows command prompt window. The title bar shows the file path: C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe. The window has a black background with white text. The first line of text is "Enter Infix Expression : (a+b)-(c*d)". The second line of text is "The Evaluated Postfix Expression is : ab+cd*-".

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
Enter Infix Expression : (a+b)-(c*d)
The Evaluated Postfix Expression is : ab+cd*-
```

Lab Program 3

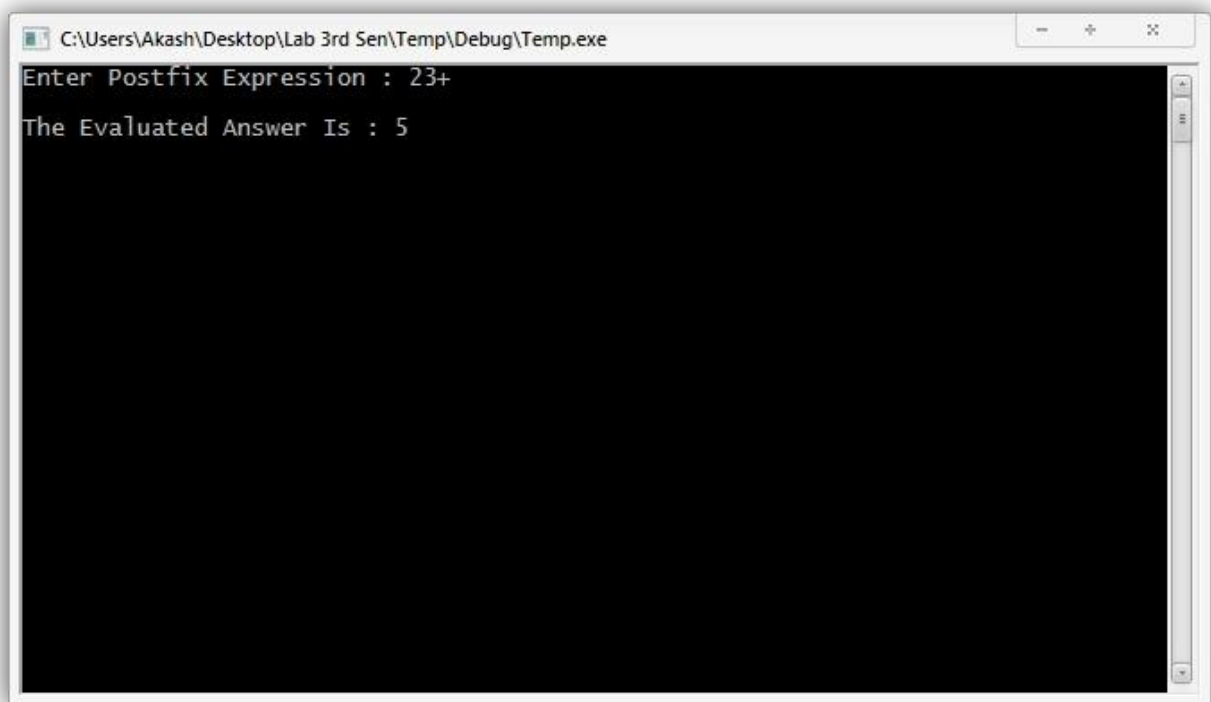
```
1  /**
2   Lab Program 3
3   Postfix Evaluation Using Stack
4   Author SkyKOG
5   */
6
7   #include <stdio.h>
8   #include <conio.h>
9   #include <string.h>
10  #include <ctype.h>
11  #include <process.h>
12
13  int stack[20];
14  int top;
15
16  void push(int);
17  int pop();
18  void eval(char post[20]);
19
20  void main()
21  {
22      char post[20];
23      top=-1;
24      printf("Enter Postfix Expression : ");
25      gets(post);
26      eval(post);
27      printf("\nThe Evaluated Answer Is : %d",pop());
28      getch();
29  }
30  void push(int ele)
31  {
32      stack[++top]=ele;
33  }
34
35  int pop()
36  {
37      return (stack[top--]);
38  }
39
40  void eval(char post[20])
41  {
42      char ch;
43      int i,b,a;
44      for(i=0;post[i];i++)
45      {
46          ch=post[i];
47          if(isdigit(ch))
48              push(ch-'0');
49          else
50          {
51              b=pop();
52              a=pop();
```

```

53         switch(ch)
54         {
55             case '+':push(a+b);
56                 break;
57             case '-':push(a-b);
58                 break;
59             case '*':push(a*b);
60                 break;
61             case '/':if(b==0)
62                 {
63                     printf("Devide By Zero");
64                     getch();
65                     exit(0);
66                 }
67                 push(a/b);
68             default:printf("Invalid Operation");
69         }
70     }
71 }
72 }

```

Output



```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
Enter Postfix Expression : 23+
The Evaluated Answer Is : 5
```

Lab Program 4

```
1  /**
2   Lab Program 4
3   Implementation of Linear Queue
4   Author SkyKOG
5   */
6
7   #include <stdio.h>
8   #include <conio.h>
9   #include <windows.h>
10  #include <process.h>
11
12  #define MAX 5
13
14  struct queue
15  {
16      int arr[MAX];
17      int f,r;
18  };
19
20  void insq(struct queue *,int);
21  void delq(struct queue *);
22  void display(struct queue *);
23
24  void main()
25  {
26      int ch,ele;
27      struct queue q;
28      q.f=-1;
29      q.r=0;
30      while(true)
31      {
32          system("cls");
33          printf("1. Insert\n2. Delete\n3. Display\n4. Exit\n\nEnter Selection : ");
34          scanf("%d",&ch);
35          switch(ch)
36          {
37              case 1:printf("\nEnter Element To Insert : ");
38                     scanf("%d",&ele);
39                     insq(&q,ele);
40                     break;
41              case 2:delq(&q);
42                     break;
43              case 3:display(&q);
44                     break;
45              case 4:exit(0);
46                     break;
47              default:printf("Invalid Selection");
48                     }
49          getch();
50      }
51  }
```

```

52 void insq(struct queue *q,int ele)
53 {
54     if(q->r==MAX)
55     {
56         printf("\nQueue Full");
57         return;
58     }
59     q->arr[q->r]=ele;
60     q->r++;
61     if(q->f==-1)
62         q->f=0;
63 }
64 void delq(struct queue *q)
65 {
66     if(q->f==-1)
67     {
68         printf("\nQueue Empty");
69         return;
70     }
71     printf("\nElement Deleted is : %d",q->arr[q->f]);
72     q->f++;
73     if(q->f==q->r)
74     {
75         q->f=-1;
76         q->r=0;
77     }
78 }
79 void display(struct queue *q)
80 {
81     int i;
82     if(q->f==-1)
83     {
84         printf("Empty Queue");
85         return;
86     }
87     printf("\nThe Contents of the Queue Are : ");
88     for(i=q->f;i<q->r;i++)
89     {
90         printf("%d ",q->arr[i]);
91     }
92 }

```


Output

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Insert
2. Delete
3. Display
4. Exit

Enter Selection : 1
Enter Element To Insert : 50
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Insert
2. Delete
3. Display
4. Exit

Enter Selection : 3
The Contents of the Queue Are : 50 40 20 10 _
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Insert
2. Delete
3. Display
4. Exit

Enter Selection : 2
Element Deleted is : 10_
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Insert
2. Delete
3. Display
4. Exit

Enter Selection : 2

Queue Empty
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Insert
2. Delete
3. Display
4. Exit

Enter Selection : 1
Enter Element To Insert : 90

Queue Full
```

Lab Program 5

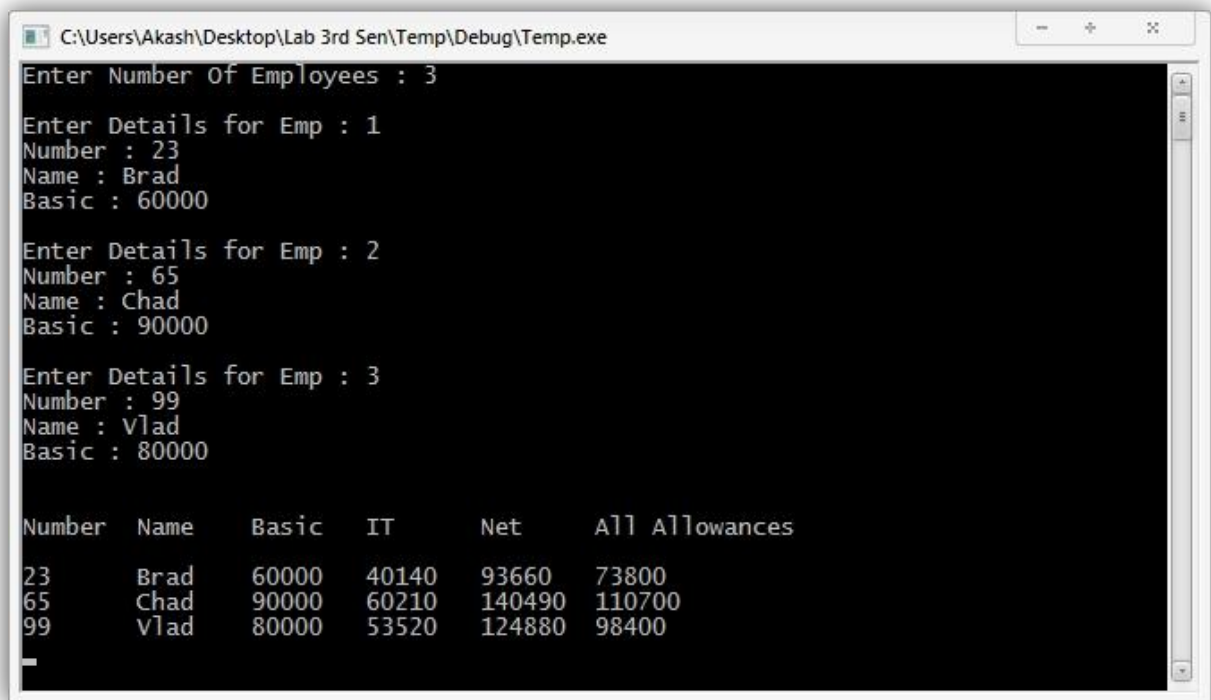
```
1  /**
2   Lab Program 5
3   Simple Class Simulation 1
4   Author SkyKOG
5   */
6
7   #include <iostream>
8   #include <Windows.h>
9   #include <conio.h>
10  #include <stdlib.h>
11
12  using namespace std;
13
14  class EMPLOYEE {
15  int Employee_Number, Basic_Salary, All_Allowances, IT, Net_Salary;
16  char Employee_Name[20];
17
18  public:
19      void accept();
20      void calc();
21      void display();
22  };
23
24  int main()
25  {
26      system("cls");
27      EMPLOYEE emp[20];
28      int n;
29      cout<<"Enter Number Of Employees : ";
30      cin>>n;
31      for(int i=0 ; i < n; i++)
32      {
33          cout<<"\nEnter Details for Emp : "<<i+1<<"\n";
34          emp[i].accept();
35      }
36
37      cout<<"\n\nNumber\tName\tBasic\tIT\tNet\tAll Allowances\n\n";
38      for(int i=0 ; i < n ; i++)
39      {
40          emp[i].display();
41      }
42
43      getch();
44      return 0;
45  }
```

```

47 void EMPLOYEE::accept()
48 {
49     cout<<"Number : ";
50     cin>>Employee_Number;
51     cout<<"Name : ";
52     cin>>Employee_Name;
53     cout<<"Basic : ";
54     cin>>Basic_Salary;
55     calc();
56 }
57
58 void EMPLOYEE::calc()
59 {
60     All_Allowances=1.23*Basic_Salary;
61     IT=0.3*(Basic_Salary+All_Allowances);
62     Net_Salary=Basic_Salary+All_Allowances-IT;
63 }
64
65 void EMPLOYEE::display()
66 {
67     cout<<Employee_Number<<"\t"<<Employee_Name<<"\t"<<Basic_Salary;
68     cout<<"\t"<<IT<<"\t"<<Net_Salary<<"\t"<<All_Allowances<<"\n";
69 }

```

Output



```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
Enter Number Of Employees : 3
Enter Details for Emp : 1
Number : 23
Name : Brad
Basic : 60000
Enter Details for Emp : 2
Number : 65
Name : Chad
Basic : 90000
Enter Details for Emp : 3
Number : 99
Name : Vlad
Basic : 80000

Number   Name    Basic   IT      Net     All Allowances
23      Brad    60000   40140   93660   73800
65      Chad    90000   60210   140490  110700
99      Vlad    80000   53520   124880  98400
```

Lab Program 6

```
1  /**
2   Lab Program 6
3   Operator Overloading & Copy Constructor - String Class
4   Author SkyKOG
5   */
6
7   #include <iostream>
8   #include <conio.h>
9   #include <Windows.h>
10  #include <string>
11
12  using namespace std;
13
14  class STRING {
15
16      char *str;
17
18      public :
19          STRING();
20          STRING(char *);
21          STRING(const STRING &);
22          STRING operator +(STRING &);
23          friend ostream & operator <<(ostream &,STRING &);
24
25  };
26
27  void main()
28  {
29      STRING s1 = "VTU";
30      STRING s2 = "BELGAUM";
31      STRING s3 = s1 + s2;
32
33      cout<<"s1 = "<<s1<<"\n";
34      cout<<"s2 = "<<s2<<"\n";
35      cout<<"s3 = "<<s3<<"\n";
36
37      getch();
38  }
39
40  STRING::STRING()
41  {
42      str=NULL;
43  }
44
45  STRING::STRING(char *x)
46  {
47      str=new char[strlen(x)+1];
48      strcpy(str,x);
49  }
```

```

51  STRING::STRING(const STRING &s3)
52  {
53      str=new char[strlen(s3.str)+1];
54      strcpy(str,s3.str);
55  }
56
57  STRING STRING::operator +(STRING &s2)
58  {
59      STRING temp;
60
61      temp.str=new char[strlen(str)+strlen(s2.str)+1];
62
63      strcpy(temp.str,str);
64      strcat(temp.str,s2.str);
65
66      return temp;
67  }
68
69  ostream & operator <<(ostream &out,STRING &s)
70  {
71      out<<s.str<<"\n\n";
72      return out;
73  }

```

Output



```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
s1 = VTU
s2 = BELGAUM
s3 = VTUBELGAUM
_
```


Lab Program 7

```
1  /**
2   Lab Program 7
3   Operator Overloading - Stack Class
4   Author SkyKOG
5   */
6
7   #include <iostream>
8   #include <conio.h>
9   #include <process.h>
10  #include <windows.h>
11
12  #define MAX 5
13
14  using namespace std;
15
16  class STACK {
17
18      int top;
19      int arr[MAX];
20
21  public:
22      STACK();
23      STACK operator + (int);
24      STACK operator -- ();
25      friend void operator <<(ostream &,STACK &);
26
27  };
28
29  void main()
30  {
31      STACK s1;
32      int ch,ele;
33
34      while(true)
35      {
36          system("cls");
37          cout<<"1. Push\n2. Pop\n3. Display\n4. Exit\n\nEnter Your Choice : ";
38          cin>>ch;
39
40          switch(ch)
41          {
42              case 1:cout<<"\nEnter The Element to Push : ";
43                  cin>>ele;
44                  s1 = s1 + ele;
45                  cout<<"\n"<<s1;
46                  break;
47              case 2:s1 = --s1;
48                  cout<<"\n"<<s1;
49                  break;
50              case 3:cout<<"\n"<<s1;
51                  break;
52              case 4:exit(0);
53                  break;
54              default:cout<<"\nInvalid Selection\n";
55          }
56          getch();
57      }
58  }
```

```

60  STACK::STACK()
61  {
62      top=-1;
63  }
64
65  STACK STACK::operator +(int ele)
66  {
67      if(top==MAX-1)
68      {
69          cout<<"\nSTACK Overflow";
70          return *this;
71      }
72
73      arr[++top]=ele;
74      return *this;
75  }
76
77
78  STACK STACK::operator --()
79  {
80      if(top==--1)
81      {
82          cout<<"\nSTACK Underflow\n";
83          return *this;
84      }
85
86      cout<<"\nThe Popped Element is : "<<arr[top--]<<"\n";
87      return *this;
88  }
89
90
91  void operator <<(ostream &out,STACK &s)
92  {
93      if(s.top==--1)
94      {
95          out<<"\nSTACK Empty";
96          return;
97      }
98
99      cout<<"\nThe Current Contents of STACK Are : ";
100     for(int i = s.top ; i >= 0 ; i--)
101     {
102         out<<s.arr[i]<<" ";
103     }
104 }

```

Output

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Push
2. Pop
3. Display
4. Exit

Enter Your Choice : 1
Enter The Element to Push : 23

The Current Contents of STACK Are : 23
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Push
2. Pop
3. Display
4. Exit

Enter Your Choice : 1
Enter The Element to Push : 77

The Current Contents of STACK Are : 77 89 56 23
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Push
2. Pop
3. Display
4. Exit

Enter Your Choice : 1
Enter The Element to Push : 85
STACK Overflow
The Current Contents of STACK Are : 99 77 89 56 23 _
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Push
2. Pop
3. Display
4. Exit

Enter Your Choice : 2

The Popped Element is : 77

The Current Contents of STACK Are : 89 56 23
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Push
2. Pop
3. Display
4. Exit

Enter Your Choice : 2

The Popped Element is : 23

STACK Empty
```

Lab Program 8

```
1  /**
2   Lab Program 8
3   Linked List
4   Author SkyKOG
5   */
6
7   #include <iostream>
8   #include <Windows.h>
9   #include <conio.h>
10
11   using namespace std;
12
13   class LIST {
14   struct node
15   {
16       int item;
17       node *link;
18   }*p;
19
20   public:
21       LIST();
22       void insert(int);
23       int del(int &);
24       void display();
25       ~LIST();
26   };
27
28   int main()
29   {
30       LIST obj;
31       int ch,ele,status=0;
32       while(true)
33       {
34           system("cls");
35           cout<<"1. Insert At Front\n2. Delete From Front\n3. Display\n4. Exit\n\nEnter Choice : ";
36           cin>>ch;|
37           switch(ch)
38           {
39               case 1:cout<<"\nEnter Element to Insert : ";
40                   cin>>ele;
41                   obj.insert(ele);
42                   break;
43               case 2:ele=obj.del(status);
44                   if(status==0)
45                   {
46                       break;
47                   }
48                   cout<<"\nElement Deleted is : "<<ele;
49                   break;
50               case 3:obj.display();
51                   break;
52               case 4:exit(0);
53                   break;
54               default:cout<<"Invalid Selection";
55                   break;
56           }getch();
57       }
58       return 0;
59   }
```

```

61 LIST::LIST()
62 {
63     p=NULL;
64 }
65
66 void LIST::insert(int item)
67 {
68     node *q;
69     q=new node;
70     q->item=item;
71     q->link=p;
72     p=q;
73 }
74
75 int LIST::del(int &status)
76 {
77     int del;
78     node *q;
79     if(p==NULL)
80     {
81         status=0;
82         cout<<"List Is Empty";
83         return NULL;
84     }
85
86     q=p;
87     status=1;
88     del=p->item;
89     p=p->link;
90     delete q;
91     return del;
92 }
93
94 void LIST::display()
95 {
96     node *q;
97     if(p==NULL)
98     {
99         cout<<"\nEmpty List";
100         return;
101     }
102     cout<<"\n\nCurrent List Status : ";
103     for(q=p ; q!=NULL ; q=q->link)
104         cout<<q->item<<" ";
105 }
106
107 LIST::~~LIST()
108 {
109     node *q;
110     if(p==NULL)
111         return;
112     while(p!=NULL)
113     {
114         q=p->link;
115         delete p;
116         p=q;
117     }
118 }

```

Output

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Insert At Front
2. Delete From Front
3. Display
4. Exit

Enter Choice : 3

Current List Status : 99 56 89 26
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Insert At Front
2. Delete From Front
3. Display
4. Exit

Enter Choice : 1
Enter Element to Insert : 100_
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Insert At Front
2. Delete From Front
3. Display
4. Exit

Enter Choice : 3

Current List Status : 100 99 56 89 26
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Insert At Front
2. Delete From Front
3. Display
4. Exit

Enter Choice : 2

Element Deleted is : 100
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Insert At Front
2. Delete From Front
3. Display
4. Exit

Enter Choice : 3

Current List Status : 99 56 89 26 _
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Insert At Front
2. Delete From Front
3. Display
4. Exit

Enter Choice : 2

List Is Empty
```


Lab Program 9

```
1  /**
2  Lab Program 9
3  Sparse Martix Search
4  Author Strider24/SkyK0G
5  */
6
7  #include <stdio.h>
8  #include <conio.h>
9
10 typedef struct
11 {
12     int row;
13     int column;
14     int value;
15 }Element;
16
17 void main()
18 {
19     Element E[10];
20     int count, key, i;
21
22
23     printf("Enter the Number of Non Zero Entities In The Sparse Matrix: ");
24     scanf("%d", &count);
25     printf("\n");
26
27     for(i = 0; i < count; i++)
28     {
29         printf("\nEnter Data For Element %d in <row,column,value> Format : ",i+1);
30         scanf("%d", &E[i].row);
31         scanf("%d", &E[i].column);
32         scanf("%d", &E[i].value);
33     }
34
35     printf("\n\nEnter the element to search for : ");
36     scanf("%d", &key);
37
38     for(i = 0; i < count; i++)
39         if(E[i].value == key)
40         {
41             printf("\nThe element %d Is Found At Row %d And Column %d", key, E[i].row, E[i].column);
42             break;
43         }
44
45     if(i == count)
46         printf("\nElement Not Found");
47
48     getch();
49 }
```

Output

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 9\Debug\Prg 9.exe
Enter the Number of Non Zero Entities In The Sparse Matrix: 4

Enter Data For Element 1 in <row,column,value> Format : 6 9 26
Enter Data For Element 2 in <row,column,value> Format : 4 5 96
Enter Data For Element 3 in <row,column,value> Format : 1 2 36
Enter Data For Element 4 in <row,column,value> Format : 698 596 1

Enter the element to search for : 96
The element 96 Is Found At Row 4 And Column 5_
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 9\Debug\Prg 9.exe
Enter the Number of Non Zero Entities In The Sparse Matrix: 3

Enter Data For Element 1 in <row,column,value> Format : 6 9 85
Enter Data For Element 2 in <row,column,value> Format : 2 3 65
Enter Data For Element 3 in <row,column,value> Format : 4 5 1

Enter the element to search for : 99
Element Not Found
```

Lab Program 10

```
1  /**
2   Lab Program 10
3   Heap Operations
4   Author SkyKOG
5   */
6
7   #include <stdio.h>
8   #include <conio.h>
9   #include <process.h>
10  #include <Windows.h>
11
12  void heaper(int,int*);
13  void insert(int,int*,int*);
14  void display(int *,int);
15
16  void main()
17  {
18      int arr[20]={1000},ele,n=0,ch;
19
20      while(true)
21      {
22          system("cls");
23          printf("1. Heap Insert\n2. Display Array\n3. Exit\n\nEnter Choice : ");
24          scanf("%d",&ch);
25          switch(ch)
26          {
27              case 1:printf("\nEnter Element to Insert : ");
28                     scanf("%d",&ele);
29                     insert(ele,arr,&n);
30                     display(arr,n);
31                     break;
32              case 2:display(arr,n);
33                     break;
34              case 3:exit(0);
35                     break;
36              default:printf("Invalid Selection");
37                     break;
38          }
39          getch();
40      }
41  }
```

```

43 void display(int *arr,int n)
44 {
45     int i;
46     printf("\n\nThe Contents of The Heap are as follows : ")
47     for(i=1;i<=n;i++)
48     {
49         printf("%d ",arr[i]);
50     }
51 }
52
53 void insert(int ele,int *arr,int *n)
54 {
55     (*n)++;
56     arr[*n]=ele;
57     heaper(*n,arr);
58 }
59
60 void heaper(int i,int *arr)
61 {
62     int val;
63     val=arr[i];
64     while(arr[i/2]<=val)
65     {
66         arr[i]=arr[i/2];
67         i=i/2;
68     }
69     arr[i]=val;
70 }

```

Output

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Heap Insert
2. Display Array
3. Exit

Enter Choice : 1
Enter Element to Insert : 8

The Contents of The Heap are as follows : 8
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Heap Insert
2. Display Array
3. Exit

Enter Choice : 1
Enter Element to Insert : 17

The Contents of The Heap are as follows : 17 8
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Heap Insert
2. Display Array
3. Exit

Enter Choice : 1
Enter Element to Insert : 63

The Contents of The Heap are as follows : 63 8 17
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Heap Insert
2. Display Array
3. Exit

Enter Choice : 1
Enter Element to Insert : 32

The Contents of The Heap are as follows : 63 32 17 8 _
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Heap Insert
2. Display Array
3. Exit

Enter Choice : 1
Enter Element to Insert : 56

The Contents of The Heap are as follows : 63 56 17 8 32
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe
1. Heap Insert
2. Display Array
3. Exit

Enter Choice : 2

The Contents of The Heap are as follows : 63 56 17 8 32
```

Lab Program 11

```
1  /**
2   Lab Program 11
3   Double Linked List Operations
4   Author SkyK06
5   */
6
7   #include <stdio.h>
8   #include <conio.h>
9   #include <process.h>
10  #include <windows.h>
11
12  struct node
13  {
14      int item;
15      struct node *l;
16      struct node *r;
17  };
18
19  struct node *getnode()
20  {
21      struct node *temp;
22      temp=(struct node *)malloc(sizeof(struct node));
23      return temp;
24  }
25
26  struct node *addatbeg(struct node *,int);
27  struct node *del(struct node *);
28  struct node *addleft(struct node *,int,int);
29  void display(struct node*);
30
31  void main()
32  {
33      struct node *p;
34      p=NULL;
35      int ele,ch,key=0;
36      while(true)
37      {
38          system("cls");
39          printf("1. Insert At Front\n2. Insert to Left\n3. Delete Node\n4. Display\n5. Exit\n\nEnter Choice : ");
40          scanf("%d",&ch);
41          switch(ch)
42          {
43              case 1:printf("\nEnter Element To Insert : ");
44                      scanf("%d",&ele);
45                      p=addatbeg(p,ele);
46                      break;
47              case 2:printf("\nEnter Element To Insert : ");
48                      scanf("%d",&ele);
49                      printf("\nEnter Key To Search On : ");
50                      scanf("%d",&key);
51                      p=addleft(p,ele,key);
52                      break;
53              case 3:p=del(p);
54                      break;
55              case 4:display(p);
56                      break;
57              case 5:exit(0);
58                      break;
59              default:printf("\nInvalid Selection");
```

```

60     }
61     getch();
62 }
63 }
64
65 void display(struct node *q)
66 {
67     struct node *r;
68     r=q;
69     if(r==NULL)
70     {
71         printf("\nList Is Empty");
72         return;
73     }
74     printf("\nThe Contents of the List Are : ");
75     while(r!=NULL)
76     {
77         printf("%d ",r->item);
78         r=r->r;
79     }
80     printf("\n");
81 }
82
83 struct node *addatbeg(struct node *q,int ele)
84 {
85     struct node *temp,*r;
86     temp=getnode();
87     temp->item=ele;
88     temp->l=NULL;
89     temp->r=NULL;
90     if(q==NULL)
91     {
92         q=temp;
93         return q;
94     }
95     temp->r=q;
96     q->l=temp;
97     q=temp;
98     return q;
99 }

```



```

101 struct node * addleft(struct node *q,int ele,int key)
102 {
103     struct node *r,*temp;
104     int flag=0;
105     temp=getnode();
106     r=q;
107     if(r==NULL)
108     {
109         printf("\nEmpty List");
110         return q;
111     }
112     if(q->item==key)
113     {
114         q=addatbeg(q,ele);
115         return q;
116     }
117     while(r->r!=NULL)
118     {
119         if(r->item==key)
120         {
121             temp->item=ele;
122             temp->r=r;
123             temp->l=r->l;
124             r->l->r=temp;
125             r->l=temp;
126             flag=1;
127         }
128         r=r->r;
129     }
130     if((r->r==NULL)&&(r->item==key))
131     {
132         temp->item=ele;
133         temp->r=r;
134         temp->l=r->l;
135         r->l->r=temp;
136         r->l=temp;
137         return q;
138     }
139
140     if(!flag)
141         printf("\nKey Not Found In List");
142     return q;
143 }

```

```

144 struct node * del(struct node *q)
145 {
146     int ele,del,status=0;
147     printf("\nEnter Element To Delete : ");
148     scanf("%d",&ele);
149     struct node *temp;
150     temp=q;
151     if(temp==NULL)
152     {
153         status=0;
154         printf("\nList Is Empty");
155         return NULL;
156     }
157
158     if(temp->item==ele)
159     {
160         status=1;
161         q=q->r;
162         del=temp->item;
163         printf("\nDeleted item is %d",del);
164         free(temp);
165         return q;
166     }
167     while(temp->r!=NULL)
168     {
169         if(temp->item==ele)
170         {
171             status=1;
172             del=temp->item;
173             printf("\nDeleted item is %d",del);
174             temp->l->r=temp->r;
175             temp->r->l=temp->l;
176             free(temp);
177             return q;
178         }
179         temp=temp->r;
180     }
181     if((temp->r==NULL)&&(temp->item==ele))
182     {
183         temp->l->r=NULL;
184         status=1;
185         del=temp->item;
186         printf("\nDeleted item is %d",del);
187         free(temp);
188         return q;
189     }
190     if(!status)
191         printf("\nElement Not in List");
192     return q;
193 }

```

Output

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 11\Debug\12. Double Link List.exe
1. Insert At Front
2. Insert to Left
3. Delete Node
4. Display
5. Exit

Enter Choice : 4

The Contents of the List Are : 80 90 56 10
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 11\Debug\12. Double Link List.exe
1. Insert At Front
2. Insert to Left
3. Delete Node
4. Display
5. Exit

Enter Choice : 1

Enter Element To Insert : 100
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 11\Debug\12. Double Link List.exe
1. Insert At Front
2. Insert to Left
3. Delete Node
4. Display
5. Exit

Enter Choice : 4

The Contents of the List Are : 100 80 90 56 10
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 11\Debug\12. Double Link List.exe
1. Insert At Front
2. Insert to Left
3. Delete Node
4. Display
5. Exit

Enter Choice : 3

Enter Element To Delete : 90

Deleted item is 90
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 11\Debug\12. Double Link List.exe
1. Insert At Front
2. Insert to Left
3. Delete Node
4. Display
5. Exit

Enter Choice : 4

The Contents of the List Are : 100 80 56 10
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 11\Debug\12. Double Link List.exe
1. Insert At Front
2. Insert to Left
3. Delete Node
4. Display
5. Exit

Enter Choice : 3

Enter Element To Delete : 99

Element Not in List
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 11\Debug\12. Double Link List.exe
1. Insert At Front
2. Insert to Left
3. Delete Node
4. Display
5. Exit

Enter Choice : 4

The Contents of the List Are : 80 56 10
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 11\Debug\12. Double Link List.exe
1. Insert At Front
2. Insert to Left
3. Delete Node
4. Display
5. Exit

Enter Choice : 2

Enter Element To Insert : 70

Enter Key To Search On : 80
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 11\Debug\12. Double Link List.exe
1. Insert At Front
2. Insert to Left
3. Delete Node
4. Display
5. Exit

Enter Choice : 4

The Contents of the List Are : 70 80 56 10
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 11\Debug\12. Double Link List.exe
1. Insert At Front
2. Insert to Left
3. Delete Node
4. Display
5. Exit

Enter Choice : 2

Enter Element To Insert : 50

Enter Key To Search On : 56
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 11\Debug\12. Double Link List.exe
1. Insert At Front
2. Insert to Left
3. Delete Node
4. Display
5. Exit

Enter Choice : 2

Enter Element To Insert : 9

Enter Key To Search On : 10
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 11\Debug\12. Double Link List.exe
1. Insert At Front
2. Insert to Left
3. Delete Node
4. Display
5. Exit

Enter Choice : 4

The Contents of the List Are : 70 80 50 56 9 10
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 11\Debug\12. Double Link List.exe
1. Insert At Front
2. Insert to Left
3. Delete Node
4. Display
5. Exit

Enter Choice : 2

Enter Element To Insert : 100

Enter Key To Search On : 999

Key Not Found In List
```

Lab Program 12

```
1  /**
2   Lab Program 7
3   Operator Overloading - date Class
4   Author SkyKOG
5   */
6
7   #include <iostream>
8   #include <conio.h>
9   #include <windows.h>
10  #include <process.h>
11
12  using namespace std;
13
14  class date {
15
16      int dd,mm,yy;
17
18      public :
19          void getdata();
20          int operator -(date);
21          date operator +(int);
22          friend ostream & operator <<(ostream &,date &);
23  };
24
25  int main()
26  {
27      int no_of_days,ch;
28      date d1,d2;
29      while(true)
30      {
31          system("cls");
32          cout<<"1. Difference Between 2 dates : \n2. Get New date : \n3. Exit\n\nEnter Selection : ";
33          cin>>ch;
34          switch(ch)
35          {
36              case 1:cout<<"\n\nEnter date d1 (dd/mm/yy) : ";
37                      d1.getdata();
38                      cout<<"Enter date d2 (dd/mm/yy) : ";
39                      d2.getdata();
40                      no_of_days=d1-d2;|
41                      if(no_of_days<0)
42                      {
43                          cout<<"\nInvalid dates";
44                      }
45                      else
46                          cout<<"\nThe Number of Days in Between Are : "<<no_of_days;
47                      break;
48              case 2:cout<<"\n\nEnter date d1 (dd/mm/yy) : ";
49                      d1.getdata();
50                      cout<<"Enter Days To Add : ";
51                      cin>>no_of_days;
52                      d2=d1+no_of_days;
53                      cout<<"\nThe Required date is : ";
54                      cout<<d2;
55                      break;
56              case 3:exit(0);
57                      break;
58              default:cout<<"\n\nInvalid Selection";
59                      break;
60          }
61          getch();
62      }
63      return 0;
64  }
```

```

66 void date::getdata()
67 {
68     cin>>dd>>mm>>yy;
69
70     if(dd>31||mm>12)
71     {
72         cout<<"\nInvalid date";
73         getdata();
74     }
75     if(((mm==4)|| (mm==6)|| (mm==9)|| (mm==11))&&(dd>30))
76     {
77         cout<<"\nInvalid date";
78         getdata();
79     }
80     if((mm==2))
81     {
82         if((yy%4!=0)&&(dd>28))
83         {
84             cout<<"\nInvalid date";
85             getdata();
86         }
87         else if((dd>29))
88         {
89             cout<<"\nInvalid date";
90             getdata();
91         }
92     }
93 }
94
95 int date::operator -(date d2)
96 {
97     int i,n1,n2,ny,d;
98     n1=n2=ny=0;
99
100    for(i=1;i<=mm;i++)
101    {
102        if((i==4)|| (i==6)|| (i==9)|| (i==11))
103            n1+=30;
104        else if(i==2)
105        {
106            if(yy%4==0)
107                n1+=29;
108            else
109                n1+=28;
110        }
111        else
112            n1+=31;
113    }
114    n1+=dd;
115
116    for(i=1;i<=d2.mm;i++)
117    {
118        if((i==4)|| (i==6)|| (i==9)|| (i==11))
119            n2+=30;
120        else if(i==2)
121        {
122            if(d2.yy%4==0)
123                n2+=29;
124            else
125                n2+=28;
126        }
127        else
128            n2+=31;
129    }
130    n2+=d2.dd;
131    ny=(yy-d2.yy)*365;
132    d=n1-n2+ny;
133    return d;

```

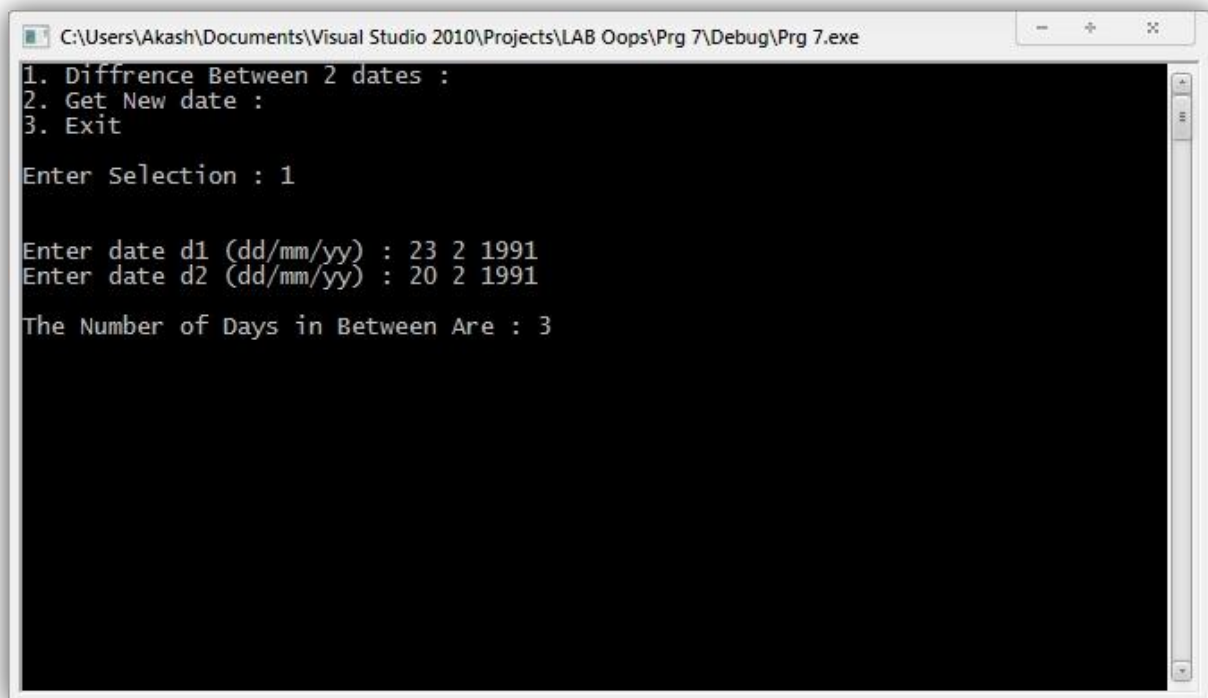


```

134 }
135
136 date date::operator +(int n)
137 {
138     while(n>365)
139     {
140         yy++;
141         if(yy%4==0)
142             n-=366;
143         n-=365;
144     }
145     while(n>30)
146     {
147         if((mm==4)||(mm==6)||(mm==9)||(mm==11))
148             n-=30;
149         else if(mm==2)
150         {
151             if(yy%4==0)
152                 n-=29;
153             else
154                 n-=28;
155         }
156         else
157             n-=31;
158         mm++;
159         if(mm>12)
160         {
161             yy++;
162             mm=1;
163         }
164     }
165     dd+=n;
166     if(dd>31)
167     {
168         if((mm==4)||(mm==9)||(mm==6)||(mm==11))
169             dd-=30;
170         else if(mm==2)
171         {
172             if(yy%4==0)
173                 dd-=29;
174             else
175                 dd-=28;
176         }
177         else
178             dd-=31;
179         mm++;
180     }
181     return *this;
182 }
183
184 ostream & operator<<(ostream &out,date &d3)
185 {
186     out<<d3.dd<<" "<<d3.mm<<" "<<d3.yy;
187     return out;
188 }

```

Output

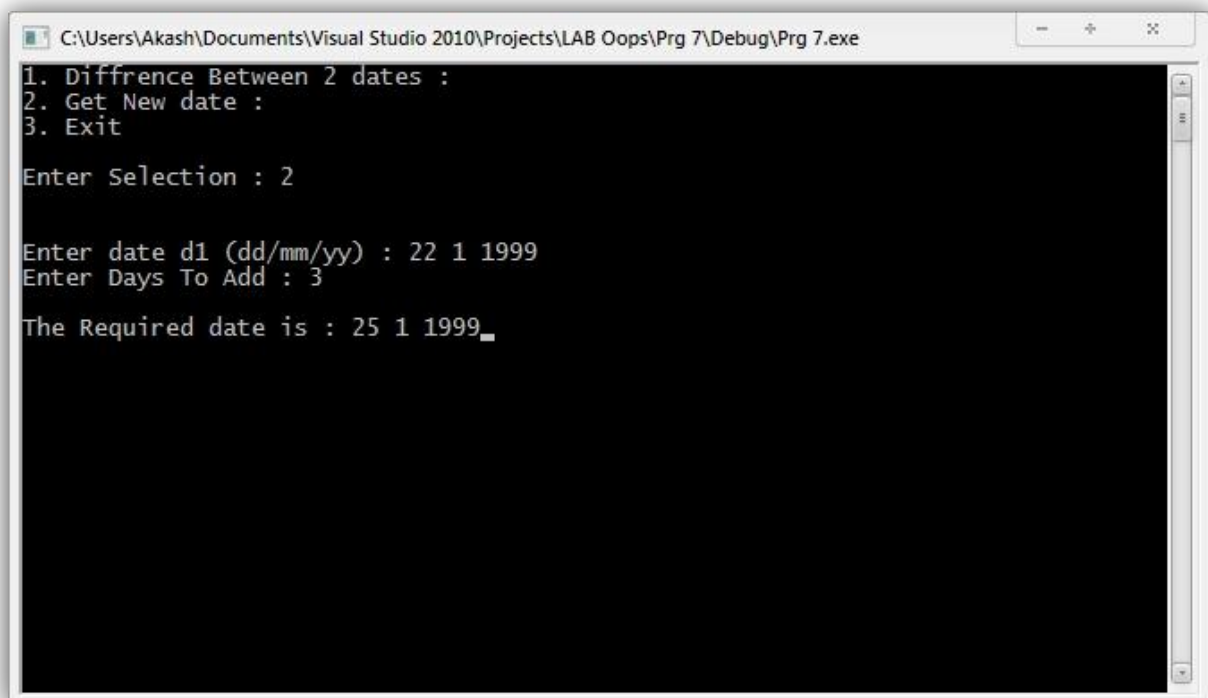


```
C:\Users\Akash\Documents\Visual Studio 2010\Projects\LAB Oops\Prg 7\Debug\Prg 7.exe
1. Difference Between 2 dates :
2. Get New date :
3. Exit

Enter Selection : 1

Enter date d1 (dd/mm/yy) : 23 2 1991
Enter date d2 (dd/mm/yy) : 20 2 1991

The Number of Days in Between Are : 3
```



```
C:\Users\Akash\Documents\Visual Studio 2010\Projects\LAB Oops\Prg 7\Debug\Prg 7.exe
1. Difference Between 2 dates :
2. Get New date :
3. Exit

Enter Selection : 2

Enter date d1 (dd/mm/yy) : 22 1 1999
Enter Days To Add : 3

The Required date is : 25 1 1999_
```

Lab Program 13

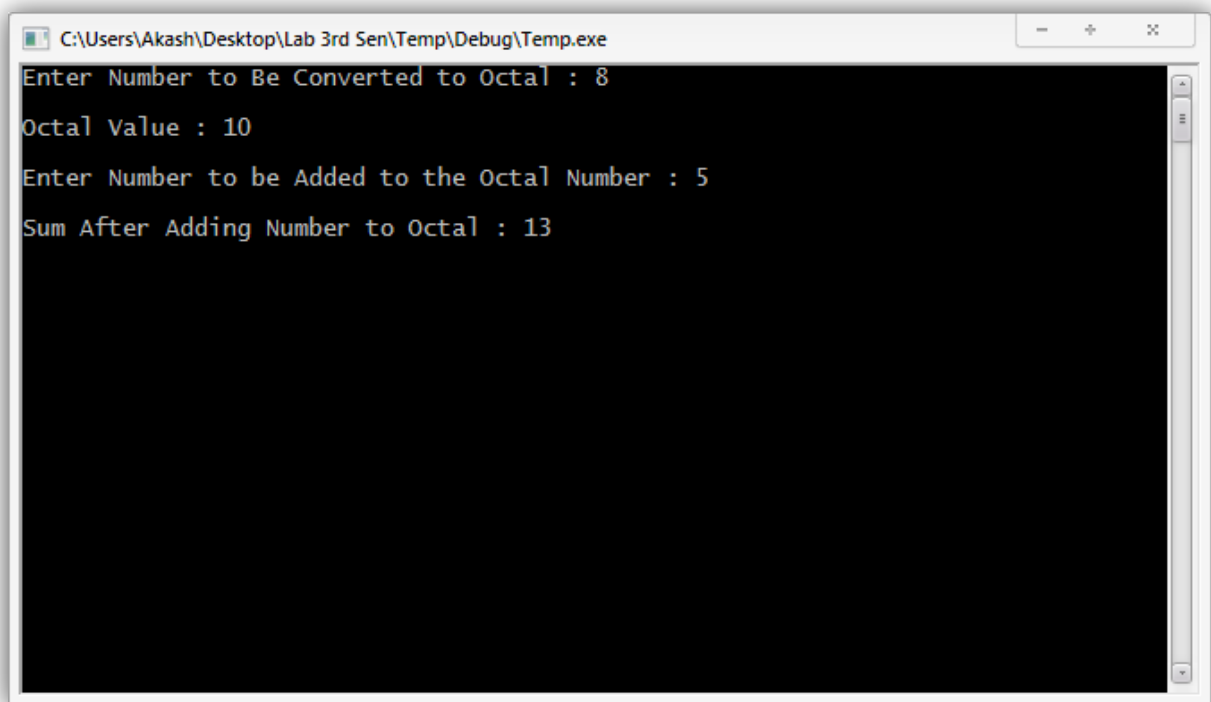
```
1  /**
2  Lab Program 13
3  Operator Overloading & Constructor - Octal Class
4  Author SkyKOG
5  */
6
7  #include <iostream>
8  #include <Windows.h>
9  #include <conio.h>
10
11  using namespace std;
12
13  class OCTAL {
14
15      int num,y;
16
17      public :
18          OCTAL(int);
19          int convert();
20          int operator +(int);
21          friend ostream & operator <<(ostream &,OCTAL &);
22
23  };
24
25  void main()
26  {
27      int x,k;
28      cout<<"Enter Number to Be Converted to Octal : ";
29      cin>>x;
30
31      OCTAL h = x;
32
33      cout<<"\nOctal Value : "<<h;
34      cout<<"\n\nEnter Number to be Added to the Octal Number : ";
35      cin>>k;
36
37      int y = h + k;
38
39      cout<<"\nSum After Adding Number to Octal : "<<y;
40      getch();
41  }
42
43
44  OCTAL::OCTAL(int x)
45  {
46      num=x;
47      y=convert();
48  }
```

```

50  int OCTAL::convert()
51  {
52      int sum=0,rem,product=1,temp=num;
53      while(temp>0)
54      {
55          rem=temp%8;
56          sum+=(product*rem);
57          product*=10;
58          temp/=8;
59      }
60      return sum;
61  }
62
63  int OCTAL::operator +(int k)
64  {
65      return num+k;
66  }
67
68  ostream & operator <<(ostream &out,OCTAL &obj)
69  {
70      out<<obj.y;
71      return out;
72  }

```

Output



A screenshot of a Windows command prompt window. The title bar shows the file path: C:\Users\Akash\Desktop\Lab 3rd Sen\Temp\Debug\Temp.exe. The window has a black background with white text. The text inside the window is as follows:

```
Enter Number to Be Converted to Octal : 8
Octal Value : 10
Enter Number to be Added to the Octal Number : 5
Sum After Adding Number to Octal : 13
```

Lab Program 14

```
1  /**
2  Lab Program 14
3  Binary Tree Operations
4  Author SkyKOG
5  */
6
7  #include <iostream>
8  #include <Windows.h>
9  #include <conio.h>
10 #include <process.h>
11
12 using namespace std;
13
14 class BIN_TREE {
15     struct node
16     {
17         node *l;
18         node *r;
19         int data;
20     }*p;
21
22     public:
23         BIN_TREE();
24         void insert(int);
25         void insertmain(node **,int);
26         void display(int);
27         void inorder(node *);
28         void preorder(node *);
29         void postorder(node *);
30
31 };
32
33 BIN_TREE::BIN_TREE()
34 {
35     p=NULL;
36 }
37
38 void main()
39 {
40     BIN_TREE obj;
41     int ch,ele;
42     while(true)
43     {
44         system("cls");
45         cout<<"1. Insert\n2. Preorder\n3. Inorder\n4. Postorder\n5. Exit\n\nEnter Selection : ";
46         cin>>ch;
47         switch(ch)
48         {
49             {
50                 case 1:cout<<"\nEnter Element To Insert : ";
51                     cin>>ele;
52                     obj.insert(ele);
53                     break;
54                 case 2:obj.display(2);
55                     break;
```

```

56         case 3:obj.display(3);
57         break;
58         case 4:obj.display(4);
59         break;
60         case 5:exit(0);
61         break;
62         default:cout<<"Invalid Selection";
63     }
64     getch();
65 }
66 }
67
68 void BIN_TREE::insert(int ele)
69 {
70     insertmain(&p,ele);
71 }
72
73 void BIN_TREE::insertmain(struct node **q,int ele)
74 {
75     if((*q)==NULL)
76     {
77         (*q)=new node;
78         (*q)->l=NULL;
79         (*q)->r=NULL;
80         (*q)->data=ele;
81         return;
82     }
83
84     else
85     {
86         if(ele==(*q)->data)
87         {
88             cout<<"Duplication Not Allowed";
89             getch();
90             return;
91         }
92         else
93         {
94             if(ele<((*q)->data))
95                 insertmain(&((*q)->l),ele);
96             else
97                 insertmain(&((*q)->r),ele);
98         }
99     }
100     return;
101 }
102

```

```

103 void BIN_TREE::display(int ch)
104 {
105     switch(ch)
106     {
107         case 2:cout<<"\nPreorder Traversal : ";
108             preorder(p);
109             break;
110         case 3:cout<<"\nInorder Traversal : ";
111             inorder(p);
112             break;
113         case 4:cout<<"\nPostorder Traversal : ";
114             postorder(p);
115             break;
116     }
117 }
118
119 void BIN_TREE::preorder(node *p)
120 {
121     if(p!=NULL)
122     {
123         cout<<p->data<<" ";
124         preorder(p->l);
125         preorder(p->r);
126     }
127 }
128
129 void BIN_TREE::inorder(node *p)
130 {
131     if(p!=NULL)
132     {
133         inorder(p->l);
134         cout<<p->data<<" ";
135         inorder(p->r);
136     }
137 }
138
139 void BIN_TREE::postorder(node *p)
140 {
141     if(p!=NULL)
142     {
143         postorder(p->l);
144         postorder(p->r);
145         cout<<p->data<<" ";
146     }
147 }

```


Output

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 14\Debug\Prg 14.exe
1. Insert
2. Preorder
3. Inorder
4. Postorder
5. Exit

Enter Selection : 1

Enter Element To Insert : 11
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 14\Debug\Prg 14.exe
1. Insert
2. Preorder
3. Inorder
4. Postorder
5. Exit

Enter Selection : 2

Preorder Traversal : 65 46 11 56 98 66 _
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 14\Debug\Prg 14.exe
1. Insert
2. Preorder
3. Inorder
4. Postorder
5. Exit

Enter Selection : 3

Inorder Traversal : 11 46 56 65 66 98 _
```

```
C:\Users\Akash\Desktop\Lab 3rd Sen\Prg 14\Debug\Prg 14.exe
1. Insert
2. Preorder
3. Inorder
4. Postorder
5. Exit

Enter Selection : 4

Postorder Traversal : 11 56 46 66 98 65 _
```