1. **Solve the recurrence C(N) = C(N/2) + N^2 for N >= 2 C(1) = 0, and N is a power of 2 [In C(N), N is the subscript; N^2 means N to power 2)**

Solution:

Given:

$C_N = C_{N/2} + N^2$ ... *(1)* $N >= 2, C_1 = 0, N=2^n$

$C_{2^n} = C_{2^n/2} + (2^n)^2$

$C_{2^n} = C_{2^{n-1}} + 2^{2n}$

Divide both sides by 2^{2n}

$(C_{2^n} / 2^{2n}) = (C_{2^{n-1}} / 2^{2n}) + 1$

$= (C_{2^{n-2}} / 2^{4n}) + 1 + 1$

...

$(C_{2^n} / 2^{2n}) = 0 + n$ $(C_{2^{n-n}} = C_{2^0} = C_1 = 0)$

$= n$

$= \lg N$ $(N=2^n$; taking lg to base 2 on both sides; $\lg N = n)$

$C_N = N^2 \lg N$ $(2^n = N \Rightarrow 2^{2n} = N^2)$

2. **Solve the recurrence C(N) = [C(N/2)]^2 for N >= 2 C(1) = 1, and N is a power of 2 [In C(N), N is the subscript)**

Solution:

Given:

$C_N = (C_{N/2})^2$ ... *(1)* $N >= 2, C_1 = 1, N=2^n$

$= (C_{N/2^1})^2$ ... same as *(1)*

$= ((C_{N/2^2})^2)^2$

$= (((C_{N/2^3})^2)^2)^2$

.....

$C_N = (C_{N/2^n})^{2...n \text{ times}}$

$= (C_{N/N})^N$ ... 2.... n times $= 2^n = N$

$= C_1$

$= 1$ ...from *(1)*

$C_N = 1$

3. **Swap**

Solution:

list.h

```
typedef struct node* link;
struct node { int item; link next; };
typedef link Node;
void initNodes(int);
link newNode(int);
void freeNode(link);
void insertNext(link, link);
```

```
link deleteNext(link);
link Next(link);
int Item(link);
```

list.c

```
#include <stdlib.h>
#include "list.h"
link freelist;
void initNodes(int N)
  {
int i;
// printf("inside initnodes\n");
   freelist = malloc((N+1)*(sizeof *freelist));
//       printf("mem allocated\n");
    for (i = 0; i < N+1; i++)
        {
               // printf("inside for\n");
                freelist[i].next = &freelist[i+1];
        }
                //printf("after next\n");
                freelist[N].next = NULL;
                 //printf("end of for\n");
  }
link newNode(int i)
  {
       link x = deleteNext(freelist);
   // printf("inside newnode\n");
       x->item = i; x->next = x;
//printf("end of new nodes\n");
           return x;

  }

void freeNode(link x)
  { insertNext(freelist, x); }

void insertNext(link x, link t)
  { t->next = x->next; x->next = t; }

link deleteNext(link x)
  { link t = x->next; x->next = t->next;
    // printf("end of delete\n");
        return t; }

link Next(link x)
  { return x->next; }

int Item(link x)
  { return x->item; }
```

client.c

```
#include "list.h"
link head;
void swap(int i, int j)
```

```
{
  link tempi,tempj, temp, previ, prevj;
  int count;
  tempi = tempj = head;
  for(count = 1; count <i;count++)
  {
        previ = tempi;
        tempi = tempi->next;
  }
  for(count = 1; count<j; count++)
  {
        prevj = tempj;
         tempj= tempj->next;
  }
  previ->next = tempj;
  prevj->next = tempi;

  temp = tempi->next;
  tempi->next = tempj->next;
  tempj->next = temp;
  if(tempi==head)
   head = tempj;
}


main()
        {
                int ith;
                int jth;
                int i;
                int j;
                int number_of_nodes;
                link temp1,temp2;
                link curr;


                printf("Please enter the number of nodes :");
                scanf("%d",&number_of_nodes);


                // assign memory locations for list
                initNodes(number_of_nodes);

                printf("Please enter item for node 1 : ");
                scanf("%d",&j);


                head = newNode(j);
                //printf("%d\n", Item(head));


                printf("Please enter item for node 2 : ");
                scanf("%d",&j);

                temp2 = newNode(j);
                //printf("%d\n", Item(temp2));
```

```
        insertNext(head,temp2);


        temp1 = temp2;

        // make new nodes and populate list
        for(i=2; i<number_of_nodes; i++)
        {
               printf("Please enter item for node %d : ",
               i+1);
                scanf("%d",&j);


                temp2 = newNode(j);
        //      printf("%d\n", Item(temp2));


                insertNext(temp1,temp2);
                temp1=temp2;

        }

// traverse and print list

temp1->next='\0';

curr = head;
printf("Printing List:\n");
 printf("%d\n",Item(curr));
while(curr->next != '\0')
{

        curr= curr->next;
        printf("%d\n",Item(curr));
}


// get user input for ith and jth elements
printf("Please enter the element to be swapped\n");
scanf("%d",&ith);
printf("Please enter the second element to be
swapped\n");
scanf("%d",&jth);


// validity check

if (ith>0 && jth>0 && ith<=number_of_nodes &&
jth<=number_of_nodes && ith != jth && ith<jth)
{
   printf("original value of first element:\n");
        temp1 = head;
        for(i=1;i<ith;i++)
                temp1= Next(temp1);
        printf("%d\n", Item(temp1));

        printf("original value of second element:\n");
```

```
                        temp1 = head;
                        for(i=1;i<jth;i++)
                                temp1= Next(temp1);
                        printf("%d\n", Item(temp1));

                        swap(ith,jth);

                        printf("swapped value of first element:\n ");
                        temp1 = head;
                        for(i=1;i<ith;i++)
                                temp1= Next(temp1);
                        printf("%d\n", Item(temp1));

                        printf("swapped value of second element:\n ");
                        temp1 = head;
                        for(i=1;i<jth;i++)
                                temp1= Next(temp1);
                        printf("%d\n", Item(temp1));
                }
                else
                {
                   printf("Enter correct values of ith and jth element.
                   It is assumed that ith is less than jth item\n");
                }

        }
```

OUTPUT FOR SWAP:

```
Script started on Fri Jan 30 14:17:01 2004
crux:~/public_html/DS/hw1_check/linked_list> swap_out

Please enter the number of nodes :6
Please enter item for node 1 : 12
Please enter item for node 2 : 23
Please enter item for node 3 : 34
Please enter item for node 4 : 45
Please enter item for node 5 : 56
Please enter item for node 6 : 67
Printing List:
12
23
34
45
56
67
Please enter the element to be swapped
3
Please enter the second element to be swapped
5
original value of first element:
34
original value of second element:
56
swapped value of first element:
 56
swapped value of second element:
```

```
 34
crux:~/public_html/DS/hw1_check/linked_list> swap_out

Please enter the number of nodes :5
Please enter item for node 1 : 123
Please enter item for node 2 : 234
Please enter item for node 3 : 345
Please enter item for node 4 : 456
Please enter item for node 5 : 567
Printing List:
123
234
345
456
567
Please enter the element to be swapped
1
Please enter the second element to be swapped
5
original value of first element:
123
original value of second element:
567
swapped value of first element:
 567
swapped value of second element:
 123
crux:~/public_html/DS/hw1_check/linked_list> swap_out

Please enter the number of nodes :4
Please enter item for node 1 : 12
Please enter item for node 2 : 23
Please enter item for node 3 : 34
Please enter item for node 4 : 45
Printing List:
12
23
34
45
Please enter the element to be swapped
2
Please enter the second element to be swapped
3
original value of first element:
23
original value of second element:
34
swapped value of first element:
 34
swapped value of second element:
 23
crux:~/public_html/DS/hw1_check/linked_list> exit

exit

script done on Fri Jan 30 14:18:30 2004
```

split.c

```
#include "list.h"
link head;
link headb;
void split(link);
void main()
{
        int ith;
        int jth;
        int i;
        int j;
        int number_of_nodes;
        link temp1,temp2;
        link curr;

        printf("Please enter the number of nodes :");
        scanf("%d",&number_of_nodes);


        // assign memory locations for list
        initNodes(number_of_nodes);

        printf("Please enter item for node 1 : ");
        scanf("%d",&j);


        head = newNode(j);
        //printf("%d\n", Item(head));


        printf("Please enter item for node 2 : ");
        scanf("%d",&j);

        temp2 = newNode(j);
        //printf("%d\n", Item(temp2));

        insertNext(head,temp2);


        temp1 = temp2;

        // make new nodes and populate list
        for(i=2; i<number_of_nodes; i++)
        {
                printf("Please enter item for node %d : ", i+1);
                scanf("%d",&j);

                temp2 = newNode(j);
                //      printf("%d\n", Item(temp2));

                insertNext(temp1,temp2);
                temp1=temp2;
        }
```

```
        // traverse and print list

        temp1->next='\0';

        curr = head;
        printf("Printing List:\n");
         printf("%d\n",Item(curr));
        while(curr->next != '\0')
        {

               curr= curr->next;
               printf("%d\n",Item(curr));
        }
         split(head);

}
void split(link heada)
{
   link temp1,temp2,curr;
   int i;
   temp1 = head;
   headb = head->next;
   temp2 = headb;
   while(temp1->next != '\0')
   {
     temp1->next = Next(temp2);
     if(temp1->next != '\0') //to check for even number of nodes
     {
        temp2->next = Next(temp1->next);
        temp1 = temp1-> next;
        temp2 = temp2->next;
    }
   }
   printf("printing first list\n");
   curr = heada;
   printf("%d\n",Item(curr));
   while( curr->next != '\0')
   {
        curr = curr->next;
         printf("%d\n",Item(curr));
   }
   printf("printing second list \n");
   curr = headb;
   printf("%d\n",Item(curr));
   while(curr->next != '\0')
   {
       curr = curr->next;
       printf("%d\n",Item(curr));
   }
}
```

SPLIT OUTPUT:

```
Script started on Fri Jan 30 14:24:05 2004
crux:~/public_html/DS/hw1_check/linked_list> split_out
```

```
Please enter the number of nodes :5
Please enter item for node 1 : 12
Please enter item for node 2 : 23
Please enter item for node 3 : 34
Please enter item for node 4 : 45
Please enter item for node 5 : 56
Printing List:
12
23
34
45
56
printing first list
12
34
56
printing second list
23
45
crux:~/public_html/DS/hw1_check/linked_list> split_out

Please enter the number of nodes :6
Please enter item for node 1 : 1
Please enter item for node 2 : 2
Please enter item for node 3 : 3
Please enter item for node 4 : 4
Please enter item for node 5 : 5
Please enter item for node 6 : 6
Printing List:
1
2
3
4
5
6
printing first list
1
3
5
printing second list
2
4
6
crux:~/public_html/DS/hw1_check/linked_list> exit

exit

script done on Fri Jan 30 14:24:52 2004
```

4. Postfix Evaluator

Solution:

STACK.h

```
typedef int Item;
void STACKinit(int);
int STACKempty();
void STACKpush(int);
```

```
Item STACKpop();
```

STACK.c

```
#include <stdlib.h>
#include "STACK.h"
static Item *s;
static int N;
void STACKinit(int maxN)
{ s = (Item *) malloc (maxN * sizeof(Item)); N = 0; }
int STACKempty()
  { return N == 0; }
void STACKpush(Item item)
  { s[N++] = item; }
Item STACKpop()
  { return s[--N]; }
```

CLIENT.c

```
#include <stdio.h>
#include <string.h>
#include "STACK.h"
main(int argc, char *argv[])
  { char *a = argv[1]; int i, tmp, N = strlen(a);
    STACKinit(N);
    for (i = 0; i < N; i++)
      {
        if (a[i] == '+')
          STACKpush(STACKpop()+STACKpop());
        if (a[i] == '*')
          STACKpush(STACKpop()*STACKpop());
       if (a[i] == '/')
        {  tmp = STACKpop();
           STACKpush(STACKpop()/tmp);
        }
        if (a[i] == '%')
        {
          tmp = STACKpop();
          STACKpush(STACKpop()%tmp);
        }
        if (a[i] == '-')
          STACKpush(-1*(STACKpop()-STACKpop()));
        if ((a[i] >= '0') && (a[i] <= '9'))
          STACKpush(0);
        while ((a[i] >= '0') && (a[i] <= '9'))
          STACKpush(10*STACKpop() + (a[i++]-'0'));
      }
    printf("%d \n", STACKpop());
}
```

SCRIPT RUN:

```
Script started on Fri Jan 30 13:46:47 2004
crux:~/public_html/DS/hw1_check/calculator> calc "60 75 +"

135
```

```
crux:~/public_html/DS/hw1_check/calculator> calc "100 50 -"

50
crux:~/public_html/DS/hw1_check/calculator> calc "20 30 *"

600
crux:~/public_html/DS/hw1_check/calculator> calc "100 50 /"

2
crux:~/public_html/DS/hw1_check/calculator> calc "100 49 %"

2

crux:~/public_html/DS/hw1_check/calculator> calc "7 5 + 6 4 - *
1 2 3 + + +"

30
crux:~/public_html/DS/hw1_check/calculator> calc "1 3 + 2 *"

8
crux:~/public_html/DS/hw1_check/calculator> calc "9 7 + 5 3 - /"

8
crux:~/public_html/DS/hw1_check/calculator> exit

exit

script done on Fri Jan 30 14:01:32 2004
```