### Experiment No. 01

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Title:** A) LCD Display Problem

B) 3n + 1 Problem

**Name:** Shivlal Arun Yadav

**Class**: TE CSE **Batch:** T1 **Roll No:** 3011

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. LCD Display Problem:

# Program:

#include <stdio.h> #include <stdlib.h> #include <math.h> #include <string.h>

int nLines; int nColumns;

char lcdNumber[23][12]; char lcdNumbers[8][23][12];

void setLcdSize(int size) { nLines = 2 \* size + 3; nColumns = size + 2;

}

void clearLcd() { int i;

for (i = 0; i < nLines; i++) { int j;

for (j = 0; j < nColumns; j++) { lcdNumber[i][j] = ' ';

}

}

}

void drawLine(int lineNumber) { int j;

for (j = 1; j < nColumns - 1; j++) { lcdNumber[lineNumber][j] = '-';

}

}

void drawFirstLine() { drawLine(0);

}

void drawMiddleLine() { drawLine(nLines / 2);

}

void drawLastLine() { drawLine(nLines - 1);

}

void drawFirstColumn(int columnNumber) { int i;

for (i = 1; i < nLines / 2; i++) { lcdNumber[i][columnNumber] = '|';

}

}

void drawLastColumn(int columnNumber) { int i;

for (i = (nLines / 2) + 1; i < nLines - 1; i++) { lcdNumber[i][columnNumber] = '|';

}

}

void drawFirstLeftColumn() { drawFirstColumn(0);

}

void drawLastLeftColumn() { drawLastColumn(0);

}

void drawFirstRightColumn() { drawFirstColumn(nColumns - 1);

}

void drawLastRightColumn() { drawLastColumn(nColumns - 1);

}

void printLcd(int position) { int i;

for (i = 0; i < nLines; i++) { int j;

for (j = 0; j < nColumns; j++) { lcdNumbers[position][i][j] = lcdNumber[i][j];

}

}

}

void printZero(int position) { clearLcd(); drawFirstLine(); drawFirstLeftColumn(); drawFirstRightColumn(); drawLastLeftColumn(); drawLastRightColumn(); drawLastLine(); printLcd(position);

}

void printOne(int position) { clearLcd(); drawFirstRightColumn(); drawLastRightColumn(); printLcd(position);

}

void printTwo(int position) { clearLcd(); drawFirstLine(); drawFirstRightColumn(); drawMiddleLine(); drawLastLeftColumn(); drawLastLine(); printLcd(position);

}

void printThree(int position) { clearLcd(); drawFirstLine(); drawFirstRightColumn(); drawMiddleLine(); drawLastRightColumn(); drawLastLine(); printLcd(position);

}

void printFour(int position) { clearLcd(); drawFirstLeftColumn(); drawFirstRightColumn(); drawMiddleLine(); drawLastRightColumn(); printLcd(position);

}

void printFive(int position) { clearLcd(); drawFirstLine(); drawFirstLeftColumn(); drawMiddleLine(); drawLastRightColumn();

drawLastLine(); printLcd(position);

}

void printSix(int position) { clearLcd(); drawFirstLine(); drawFirstLeftColumn(); drawMiddleLine(); drawLastLeftColumn(); drawLastRightColumn(); drawLastLine(); printLcd(position);

}

void printSeven(int position) { clearLcd(); drawFirstLine(); drawFirstRightColumn(); drawLastRightColumn(); printLcd(position);

}

void printEight(int position) { clearLcd(); drawFirstLine(); drawFirstLeftColumn(); drawFirstRightColumn(); drawMiddleLine(); drawLastLeftColumn(); drawLastRightColumn(); drawLastLine(); printLcd(position);

}

void printNine(int position) { clearLcd(); drawFirstLine(); drawFirstLeftColumn(); drawFirstRightColumn(); drawMiddleLine(); drawLastRightColumn(); drawLastLine(); printLcd(position);

}

void printNumber(int n, int position) {

void (\*functions[10]) (int position) = { printZero,

printOne, printTwo, printThree, printFour,

printFive, printSix, printSeven, printEight, printNine

};

functions[n](position);

}

int main() {

int size = 0;

char strNumber[9];

scanf("%d %s", &size, strNumber);

while (size != 0 || atoi(strNumber) != 0) { setLcdSize(size);

int length = strlen(strNumber);

int i;

for (i = 0; i < length; i++) { int n = strNumber[i] - '0'; printNumber(n, i);

}

for (i = 0; i < nLines; i++) { int n;

for (n = 0; n < length; n++) { int j;

for (j = 0; j < nColumns; j++) { printf("%c", lcdNumbers[n][i][j]);

}

if (n < length - 1) {

printf(" "); //column of blanks

}

}

printf("\n");

}

printf("\n");

scanf("%d %s", &size, strNumber);

}

return 0;

}

# Output:

**2 1234**

**-- --**

|  |  |  |
| --- | --- | --- |
| **| |** | **| |** | **|** |
| **| |** | **| |** | **|** |
| **--** | **--** | **--** |
| **| |** | **|** | **|** |
| **| |** | **|** | **|** |
| **--** | **--** |  |

1. 3n + 1 Problem

# Program:

#include <stdio.h> #include <string.h>

#define max(x, y) ((x) > (y) ? (x) : (y))

#define swap(x, y) {int tmp; tmp = x, x = y, y = tmp;} #define N 1000000

#define node 1048576

int A[N+1], tree[node<<1];

void setTree(int l, int r, int k) { if(l == r)

tree[k] = A[l]; if(l < r) {

int m = (l+r) >> 1; setTree(l, m, k<<1); setTree(m+1, r, (k<<1)+1);

tree[k] = max(tree[k<<1], tree[(k<<1)+1]);

}

}

int getTree(int l, int r, int k, int i, int j) { if(l == i && r == j)

return tree[k]; if(l < r) {

int m = (l+r) >> 1; if(m >= j)

return getTree(l, m, k<<1, i, j); else if(m < i)

return getTree(m+1, r, (k<<1)+1, i, j); else

return max(getTree(l, m, k<<1, i, m), getTree(m+1, r, (k<<1)+1,

m+1, j));

}

}

void build() { long long n; int i, len;

memset(A, 0, sizeof(A));

memset(tree, 0, sizeof(tree));

A[1] = 1;

for(i = 2; i <= N; i++) { n = i, len = 1; while(n != 1) {

if(n&1) {

n = n\*3 + 1;

} else {

n >>= 1;

}

if(n < N && A[n] != 0) { len += A[n];

break;

}

len ++;

}

A[i] = len;

}

setTree(1, N, 1);

}

int main() {

build(); int i, j;

while(scanf("%d %d", &i, &j) == 2) { printf("%d %d ", i, j);

if(i > j)

swap(i, j);

printf("%d\n", getTree(1, N, 1, i, j));

}

return 0;

}

**Output:**

1 10

1 10 20

|  |  |
| --- | --- |
| 100 | 200 |
| 100 | 200 125 |
| 201 | 210 |
| 201 | 210 89 |
| 900 | 1000 |
| 900 | 1000 174 |

### Experiment No. 02

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Title:** A) The Trip.

B) Australian Voting.

**Name:** Shivlal Arun Yadav

**Class**: TE CSE **Batch:** T1 **Roll No:** 3011

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. The Trip.

# Program:

#include <iostream> #include <iomanip> #include <cmath> using namespace std; int students[1005]; int main()

{

int N;

while (cin>> N, N)

{

int sum = 0;

for (inti = 0; i< N; ++i)

{

int dollar, cent; char temp;

cin>> dollar >> temp >> cent; students[i] = dollar \* 100 + cent; sum += students[i];

}

intlowAverage = sum / N;

inthighAverage = lowAverage + ((sum % N) ? 1 : 0); intsumAbove = 0;

for (inti = 0; i< N; ++i)

{

if (students[i] >highAverage)

sumAbove += students[i] - highAverage;

}

intsumBelow = 0;

for (inti = 0; i< N; ++i)

{

if (students[i] <lowAverage)

sumBelow += lowAverage - students[i];

}

intusedSum = max(sumAbove, sumBelow);

cout<<'$'<< (usedSum / 100) <<'.'<<setw(2) <<setfill('0') << (usedSum % 100)

<<'\n';

}

}

**Output:**



1. **Australian Voting Program:**

#include <iostream> #include <vector> #include <sstream> using namespace std; int main()

{

int T; cin>> T;

stringendingSeperator = ""; while (T--)

{

int n; cin>> n;

vector<string> names(n); vector<bool> eliminated(n, false); cin.ignore();

for (inti = 0; i< n; ++i) getline(cin, names[i]); string temp; getline(cin, temp);

vector<vector<int>> ratings; while (temp != "")

{

stringstreamss; ss<< temp;

vector<int> order(n); for (inti = 0; i< n; ++i)

{

ss>> order[i];

--order[i];

}

ratings.push\_back(order); if (cin.eof())

break;

getline(cin, temp);

}

intnumRatings = ratings.size(); vector<int>posInRatings(numRatings, 0); int winner = -1;

vector<int> count(n, 0);

for (inti = 0; i<numRatings; ++i)

++count[ratings[i][0]]; while (winner == -1)

{

/\*

cout<<"Current eliminated:\n"; for (inti = 0; i<n; ++i)

cout<< eliminated[i] <<""; cout<<'\n';

\*/

for (inti = 0; i<numRatings; ++i)

{

bool changed(false);

while (eliminated[ratings[i][posInRatings[i]]])

{

++posInRatings[i]; changed = true;

}

if (changed)

++count[ratings[i][posInRatings[i]]];

}

/\*

cout<<"Current vote count:\n"; for (inti = 0; i< n; ++i)

cout<< count[i] <<""; cout<<'\n';

cout<<"Current pos in Ratings:\n"; for (inti = 0; i<numRatings; ++i) cout<<posInRatings[i] <<""; cout<<'\n';

\*/

int highest(0); int lowest(1000);

for (inti = 0; i< n; ++i)

{

if (eliminated[i]) continue;

if (count[i] > highest) highest = count[i];

if (count[i] < lowest) lowest = count[i];

}

if (highest == lowest || highest \* 2 >numRatings) winner = highest;

else

{

for (inti = 0; i< n; ++i) if (count[i] == lowest) eliminated[i] = true;

}

}

/\*

cout<<"Current vote count:\n"; for (inti = 0; i< n; ++i)

cout<< count[i] <<""; cout<<'\n';

cout<<"Winner has "<< winner <<" votes. ";

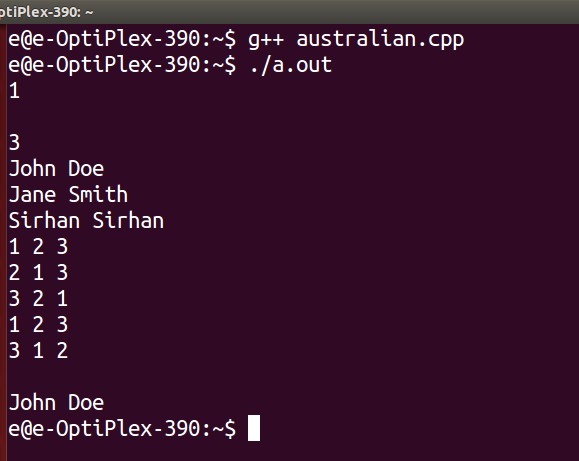
\*/ cout<<endingSeperator; endingSeperator = "\n"; for(inti = 0; i< n; ++i)

if (count[i] == winner && !eliminated[i]) cout<< names[i] <<'\n';

}

}

# Output:



### Experiment No. 03

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Title:** A) Crypt Cracker.

B) Jolly Jump.

**Name:** Shivlal Arun Yadav

**Class**: TE CSE **Batch:** T1 **Roll No:** 3011

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### Crypt Cracker.

**Program:**

#include<stdio.h> #include<cstring> #include<set> #include<algorithm> using namespace std; struct data

{

int id,pal[11],solve[11],tl\_pal,sub; set<int>s;

};

bool comp(data a,data b)

{

if(a.s.size()==b.s.size())

{

if(a.tl\_pal==b.tl\_pal) return a.id<b.id;

return (a.tl\_pal<b.tl\_pal);

}

return (a.s.size()>b.s.size());

}

data vec[102]; int main()

{

//freopen ("in.txt", "r", stdin); int t,i,j,k,l;

int id,prb,pal; char ver,ar[250]; scanf("%d\n",&t); while(t--)

{

for(i=0; i<101; i++)

{

memset(vec[i].pal,0,sizeof(vec[i].pal)); memset(vec[i].solve,0,sizeof(vec[i].solve)); vec[i].tl\_pal=0;

vec[i].sub=0;

vec[i].s.clear();

}

while(gets(ar) && ar[0]!='\0')

{

sscanf(ar,"%d %d %d %c",&id,&prb,&pal,&ver); vec[id].id=id;

vec[id].sub++; if(ver=='C')

{

if(!vec[id].solve[prb])

{

vec[id].solve[prb]=1; vec[id].tl\_pal+=pal+(vec[id].pal[prb]\*20); vec[id].s.insert(prb);

}

}

else if(ver=='I' && vec[id].solve[prb]==0) vec[id].pal[prb]++;

}

sort(vec,vec+101,comp); for(i=0; i<101; i++)

{

if(vec[i].sub>0)

printf("%d %d %d\n",vec[i].id,vec[i].s.size(),vec[i].tl\_pal);

}

if(t>0)

printf("\n");

}

return 0;

}

# Output:

6

and dick jane puff spot yertle

bjvg xsb hxsn xsb qymm xsb rqat xsb pnetfn xxxx yyy zzzz www yyyy aaa bbbb ccc dddddd

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Dick | and | jane | and | puff | and | spot | and | yertle |
| \*\*\*\* | \*\*\* | \*\*\*\* | \*\*\* | \*\*\*\* | \*\*\* | \*\*\*\* | \*\*\* | \*\*\*\*\*\* |

### Jolly Jump.

**Program:**

#include <cstdio> #include <cmath> #include <set>

using namespace std; int main() {

int n, m, l;

while (scanf("%d", &n) != EOF) { set<int> checker;

bool jolly = true;

scanf("%d", &m);

for (int i = 0; i < n - 1; i++) { scanf("%d", &l);

m = abs(m - l); checker.insert(m); m = l;

}

if (checker.size() != n - 1) jolly = false;

if (checker.count(0) > 0)

jolly = false;

if (checker.upper\_bound(n - 1) != checker.end()) jolly = false;

if (jolly)

printf("Jolly\n");

else

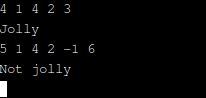
}

printf("Not jolly\n");

return 0;

}

# Output:



### Experiment No. 04

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Title:** A) Crypt Cracker.

B) Contest Scoreboard.

**Name:** Shivlal Arun Yadav

**Class**: TE CSE **Batch:** T1 **Roll No:** 3011

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. Stack 'em Up

# Program:

#include <cstdio> #include <string> #include <iostream> #include <algorithm> using namespace std; int main()

{

// Are stored [num % 13][num / 13]

string cardFirst[] = {"2", "3", "4", "5", "6", "7", "8", "9", "10", "Jack", "Queen", "King", "Ace" };

string cardSecond[] = {"Clubs", "Diamonds", "Hearts", "Spades"}; string temp;

char separator[2];

separator[0] = separator[1] = '\0';

int shuffles[101][52], numShufflesKnown, numShufflesDone, orderOfShuffles[101], T, pos;

scanf("%d", &T); while (T--)

{

printf("%s", separator); separator[0] = '\n';

scanf("%d", &numShufflesKnown);

for (int i = 0; i < numShufflesKnown; ++i)

{

for (int j = 0; j < 52; ++j)

{

scanf("%d", &shuffles[i][j]);

--shuffles[i][j];

}

}

cin.ignore(); numShufflesDone = 0; getline(cin, temp); while (temp != "")

{

orderOfShuffles[numShufflesDone++] = atoi(temp.c\_str()) - 1; getline(cin, temp);

}

for (int i = 0; i < 52; ++i)

{

pos = i;

for (int c = numShufflesDone - 1; c >= 0; --c) pos = shuffles[orderOfShuffles[c]][pos];

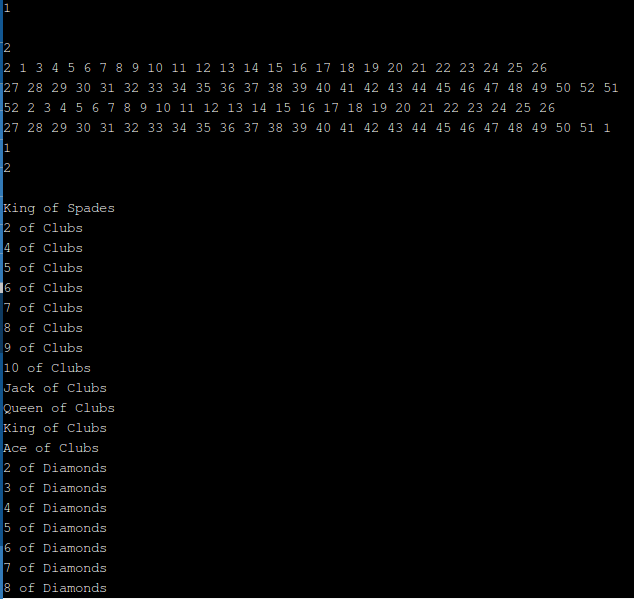
printf("%s of %s\n", cardFirst[pos % 13].c\_str(), cardSecond[pos / 13].c\_str());

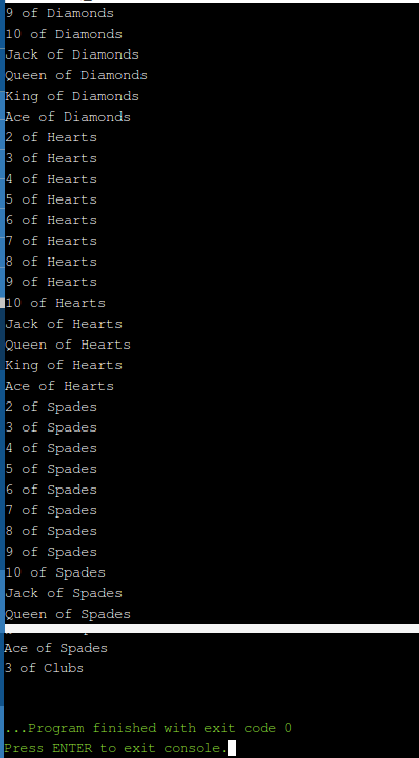
}

}

}

### Output:





1. **Contest Scoreboard.**

**Program:**

#include<stdio.h> #include<cstring> #include<set> #include<algorithm>

using namespace std; struct data

{

int id,pal[11],solve[11],tl\_pal,sub; set<int>s;

};

bool comp(data a,data b)

{

if(a.s.size()==b.s.size())

{

if(a.tl\_pal==b.tl\_pal) return a.id<b.id;

return (a.tl\_pal<b.tl\_pal);

}

return (a.s.size()>b.s.size());

}

data vec[102]; int main()

{

//freopen ("in.txt", "r", stdin); int t,i,j,k,l;

int id,prb,pal; char ver,ar[250]; scanf("%d\n",&t); while(t--)

{

for(i=0; i<101; i++)

{

memset(vec[i].pal,0,sizeof(vec[i].pal)); memset(vec[i].solve,0,sizeof(vec[i].solve)); vec[i].tl\_pal=0;

vec[i].sub=0; vec[i].s.clear();

}

while(gets(ar) && ar[0]!='\0')

{

sscanf(ar,"%d %d %d %c",&id,&prb,&pal,&ver); vec[id].id=id;

vec[id].sub++; if(ver=='C')

{

if(!vec[id].solve[prb])

{

vec[id].solve[prb]=1; vec[id].tl\_pal+=pal+(vec[id].pal[prb]\*20); vec[id].s.insert(prb);

}

}

else if(ver=='I' && vec[id].solve[prb]==0) vec[id].pal[prb]++;

}

// for(i=0;i<9;i++)

// printf("%d ",vec[1].pal[i]);

// printf("\n"); sort(vec,vec+101,comp); for(i=0; i<101; i++)

{

if(vec[i].sub>0)

printf("%d %d %d\n",vec[i].id,vec[i].s.size(),vec[i].tl\_pal);

}

if(t>0)

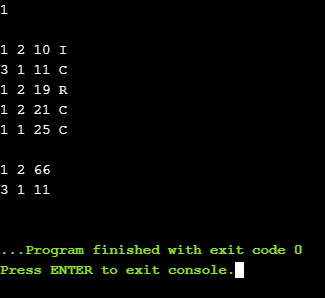
printf("\n");

}

return 0;

}

# Output:



### Experiment No. 05

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Title:** A) WERTYU

B) Common Prmutation.

**Name:** Shivlal Arun Yadav

**Class**: TE CSE **Batch:** T1 **Roll No:** 3011

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### WERTYU

**Program:**

#include <cstdio> using namespace std; char translate(char c) {

switch (c) { case '1':

return '`'; case '2':

return '1';

case '3':

return '2';

case '4':

return '3';

case '5':

return '4';

case '6':

return '5';

case '7':

return '6';

case '8':

return '7';

case '9':

return '8';

case '0':

return '9';

case '-':

return '0';

case '=':

return '-';

case 'W':

return 'Q'; case 'E':

return 'W'; case 'R':

return 'E'; case 'T':

return 'R'; case 'Y':

return 'T';

case 'U':

return 'Y';

case 'I':

return 'U';

case 'O':

return 'I'; case 'P':

return 'O';

case '[':

case ']':

return 'P'; return '[';

case '\\':

return ']';

case 'S':

return 'A'; case 'D':

return 'S'; case 'F':

return 'D'; case 'G':

return 'F'; case 'H':

return 'G';

case 'J':

return 'H';

case 'K':

return 'J'; case 'L':

return 'K';

case ';':

return 'L';

case '\'':

return ';';

case 'X':

return 'Z'; case 'C':

return 'X'; case 'V':

return 'C'; case 'B':

return 'V'; case 'N':

return 'B'; case 'M':

return 'N';

case ',':

case '.':

return 'M'; return ',';

case '/':

case ' ':

return '.';

return ' ';

case '\n':

return '\n';

}

return ' ';

}

int main() {

char c;

while (scanf("%c", &c) != EOF) {

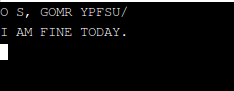
printf("%c", translate(c));

}

return 0;

}

# Output:



## Common Prmutation

**Program:**

#include <iostream> #include <string> #include <algorithm> using namespace std; int main()

{

string a, b;

while (getline(cin, a), getline(cin, b))

{

int aCounts[128], bCounts[128]; for (int i = 'a'; i <= 'z'; ++i)

aCounts[i] = bCounts[i] = 0; for (int i = 0; i < a.size(); ++i)

++aCounts[a[i]];

for (int i = 0; i < b.size(); ++i)

++bCounts[b[i]];

string longest = "";

for (int i = 'a'; i <= 'z'; ++i)

while (aCounts[i] > 0 && bCounts[i] > 0)

{

longest += (char) i;

--aCounts[i];

--bCounts[i];

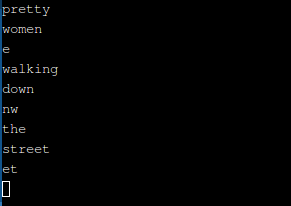
}

cout << longest << '\n';

}

}

# Output:



### Experiment No. 06

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Title:** A) Doublets

B) File Fragmentation.

**Name:** Shivlal Arun Yadav

**Class**: TE CSE **Batch:** T1 **Roll No:** 3011

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### Doublets

**Program:**

#include <set> #include <map> #include <list> #include <cmath> #include <ctime> #include <climits> #include <queue> #include <stack> #include <cctype> #include <cstdio> #include <string> #include <vector> #include <cassert> #include <cstdlib> #include <cstring> #include <sstream> #include <iostream> #include <algorithm> using namespace std;

#define FOR(i, L, U) for(int i=(int)L; i<=(int)U; i++) #define FORD(i, U, L) for(int i=(int)U; i>=(int)L; i--) #define READ(x) freopen(x, "r", stdin)

#define WRITE(x) freopen(x, "w", stdout) #define PQ priority\_queue

#define PB push\_back #define SZ size() #define ff first #define ss second #define EPS 1e-9

#define SQR(x) ((x)\*(x)) #define INF 99999999

#define ALL\_BITS ((1 << 31) - 1)

#define NEG\_BITS(mask) (mask ^= ALL\_BITS) #define TEST\_BIT(mask, i) (mask & (1 << i)) #define ON\_BIT(mask, i) (mask |= (1 << i))

#define OFF\_BIT(mask, i) (mask &= NEG\_BITS(1 << i)) typedef long long LL;

typedef vector<char> VC;

typedef vector<vector<char> > VVC; typedef vector<int> VI;

typedef vector<vector<int> > VVI; typedef vector<string> VS; typedef vector<bool> VB;

typedef vector< vector<bool> > VVB; typedef pair<int, int> PII;

typedef map<int, int> MII; typedef map<char, int> MCI; typedef map<string, int> MSI; typedef map<int, string> MIS; #define WHITE 0

#define GRAY 1

#define BLACK 2

#define MAX\_NODE 25145 string name;

int nodes;

int dist[MAX\_NODE]; bool color[MAX\_NODE]; int pre[MAX\_NODE];

int u,v;

MIS rev;

MSI dic;

void bfs(int src){

map<int,string> ::iterator revEnd = rev.end(); string ustr,vstr;

queue<int> q;

FOR(i,1,nodes){

dist[i] = INF; color[i] = false; pre[i] = i;

}

dist[src] = 0; color[src] = true; q.push(src); while(!q.empty()){

u = q.front(); ustr = rev[u]; q.pop();

FOR(i,0,ustr.length()-1){

FOR(j,'a','z'){

if(ustr[i]==j)continue; vstr= ustr;

vstr[i] = j;

v = dic[vstr]; if(color[v]==false&&rev.find(v)!=revEnd){

color[v] = true; dist[v] = dist[u] + 1; pre[v] = u;

q.push(v);

}

}

}

}

}

int main()

{

//READ("input.txt");

//WRITE("output.txt"); map<string,int> ::iterator it; string str;

string st,en;

bool letBlank = false; while(getline(cin,name)){

if(name!=""){

if(!dic[name]){ dic[name] = ++nodes; rev[nodes] = name;

}

}

else while(cin >> st >> en){ if(letBlank) printf("\n"); letBlank = true; bfs(dic[st]);

if(dist[dic[en]]==INF)printf("No solution.\n"); else {

VI path; int u,v;

v = dic[en]; u = dic[st]; while(v!=u){

path.push\_back(v); v = pre[v];

}

path.push\_back(u);

FORD(i,path.size()-1,0) cout << rev[path[i]] << endl;

}

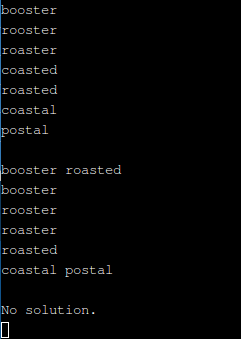
}

}

return 0;

}

**Output:**



* 1. **File Fragmentation Program:**

#include <iostream> #include <sstream> #include <map> #include <string> #include <vector> using namespace std; int main()

{

string s; getline(cin, s); size\_t T; istringstream ss(s); ss >> T;

// Skip the first empty line. getline(cin, s);

while ( T-- )

{

vector<string> fragments;

while (getline(cin, s) && !s.empty())

{

ss.clear();

ss.str(s);

string fragment; ss >> fragment;

fragments.push\_back(fragment);

}

// Consider all concatenations of any two strings. map<string, int> memo;

for (size\_t i = 0; i < fragments.size(); ++i)

for (size\_t j = i + 1; j < fragments.size(); ++j)

{

++memo[fragments[i] + fragments[j]];

++memo[fragments[j] + fragments[i]];

}

// Search for the string of highest count. map<string, int>::iterator iter(memo.begin()); map<string, int>::iterator file(memo.begin()); for (; iter != memo.end(); ++iter)

{

if (iter->second > file->second) file = iter;

}

cout << file->first << endl; if (T)

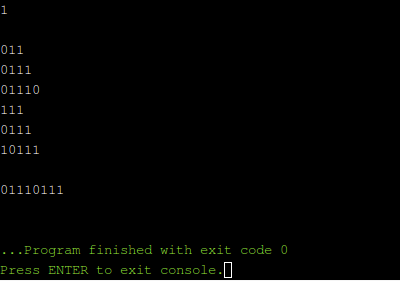
cout << endl;

}

return 0;

}

# Output:



### Experiment No. 07

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Title:** A) Vito’s Family.

B) Shoemaker’s Problem

.

**Name:** Shivlal Arun Yadav

**Class**: TE CSE **Batch:** T1 **Roll No:** 3011

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### Vito’s Family Program:

#include <cstdio> #include <algorithm> using namespace std; int main() {

int tc, r, median, x[505], ans; scanf("%d", &tc);

while (tc--) {

scanf("%d", &r);

for (int i = 0; i < r; i++) { scanf("%d", &x[i]);

}

sort(x, x + r); median = x[r / 2]; ans = 0;

for (int i = 0; i < r; i++) {

ans += abs(median - x[i]);

}

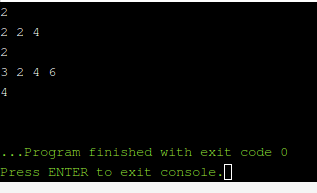
printf("%d\n", ans);

}

return 0;

}

**Output:**



1. **Shoemaker’s Problem Program:**

#include <iostream>

#include <vector> #include<string> #include<algorithm> #include<queue> using namespace std; struct shoe

{

int time, cost, name;

};

int main()

{

int cases, num, x, y, z; cin>>cases; vector<shoe> vec; vector<int> retVec;

for (x=0; x<cases; x++)

{

vec.clear(); retVec.clear(); cin.ignore(255,'\n'); string buff; getline(cin,buff); int totalTime=0; cin>>num;

for (y=0; y<num; y++)

{

shoe s; cin>>s.time>>s.cost; s.name=y+1; totalTime+=s.time; vec.push\_back(s);

}

int worst; int wNum;

for (y=0; y<vec.size();)

{

worst=-1;

for (z=0; z<vec.size(); z++)

{

if (vec[z].cost\*(totalTime-vec[z].time)>worst)

{

worst=vec[z].cost\*(totalTime-vec[z].time); wNum=z;

}

}

retVec.push\_back(vec[wNum].name); totalTime-=vec[wNum].time; vec.erase(vec.begin()+wNum);

}

for (y=0; y<retVec.size(); y++)

{

cout<<retVec[y];

if (y!=retVec.size()-1) cout<<" ";

}

cout<<endl;

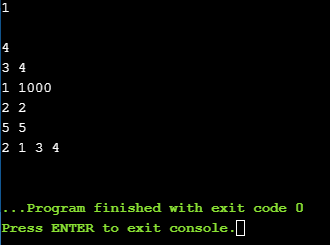
if (x!=cases-1) cout<<endl;

}

return 0;

}

## Output:



### Experiment No. 08

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Title:** A) Shell Sort

B) Football

.

**Name:** Shivlal Arun Yadav

**Class**: TE CSE **Batch:** T1 **Roll No:** 3011

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## Shell Sort

**Program:**

#include <stdio.h> #include <string.h> int main() {

int t, n, i, j;

char org[210][88], des[210][88]; scanf("%d", &t);

while(t--) {

scanf("%d", &n); getchar();

for(i = 0; i < n; i++) gets(org[i]);

for(i = 0; i < n; i++) gets(des[i]);

for(i = n-1, j = n-1; i >= 0; i--)

if(!strcmp(org[i], des[j])) j--;

for(; j >= 0; j--)

puts(des[j]);

puts("");

}

return 0;

}

# Output:



## Football Program:

#include<bits/stdc++.h>

#define CLR(a,b) memset(a, b, sizeof(a)) using namespace std;

map<string , int> teams; map<string , int> :: iterator it;

int team\_goal\_dise[100], team\_goal\_khaise[100], total\_khelse[100]; int winner[100], point[100], looser[100], draw[100];

struct data{ string teamname;

int point, goaldise, goal\_diff, khelse, jitse;

};

bool cmp(data one, data two)

{

char bread[105], butter[105]; if(one.point==two.point)

{

if(one.jitse==two.jitse)

{

if(one.goal\_diff==two.goal\_diff)

{

if(one.goaldise == two.goaldise)

{

if(one.khelse==two.khelse)

{

int szbread = one.teamname.size(); int szbutter = two.teamname.size(); for(int i=0; i<szbread; i++)

bread[i] = tolower(one.teamname[i]); for(int i=0; i<szbutter; i++)

butter[i] = tolower(two.teamname[i]); return strcmp(bread, butter)<0;

}

else return one.khelse<two.khelse;

}

else return one.goaldise>two.goaldise

}

else return one.goal\_diff>two.goal\_diff;

}

else return one.jitse>two.jitse;

}

else return one.point>two.point;

}

int main()

{

int test, n, m, goalone, goaltwo;

string name, score, adi1, adi2, tourn\_name; scanf("%d", &test);

getchar(); while(test--)

{

getline(cin,tourn\_name);

//getchar(); teams.clear();

CLR(team\_goal\_dise, 0); CLR(team\_goal\_khaise, 0); CLR(looser,0);

CLR(total\_khelse, 0); CLR(winner, 0); CLR(point, 0); CLR(draw, 0); scanf("%d", &n);

getchar();

for(int i=0; i<n; i++)

{

getline(cin,name);

// cout<<name<<endl; teams[name] = i;

}

scanf("%d",&m); getchar();

for(int i=0; i<m; i++)

{

getline(cin,score);

//cout<<score<<endl; int sz = score.size(), j; goalone = 0; adi1 = ""; bool hashfound = false;

for(j=0; score[j]!='@'; j++)

{

if(score[j]=='#') { hashfound = true; continue; } if(!hashfound)

{

adi1 = adi1 + score[j];

}

else if(isdigit(score[j]))

{

goalone = goalone\*10 + (score[j]-'0');

}

}

adi2 ="";goaltwo = 0; hashfound = false; for(int l=j+1; score[l]!='\0'; l++)

{

if(score[l]=='#') {hashfound = true; continue; } if(hashfound)

{

adi2 = adi2 + score[l];

}

else if(isdigit(score[l]))

{

goaltwo = goaltwo\*10 + (score[l]-'0');

}

}

//cout<<goalone<<' '<<adi1<<' '<<goaltwo<<' '<<adi2<<endl; team\_goal\_dise[teams[adi1]] = team\_goal\_dise[teams[adi1]] + goalone; team\_goal\_dise[teams[adi2]] = team\_goal\_dise[teams[adi2]] + goaltwo; team\_goal\_khaise[teams[adi1]] = team\_goal\_khaise[teams[adi1]] + goaltwo;

team\_goal\_khaise[teams[adi2]] = team\_goal\_khaise[teams[adi2]] + goalone; total\_khelse[teams[adi1]]++;

total\_khelse[teams[adi2]]++; if(goalone>goaltwo)

{

winner[teams[adi1]]++; looser[teams[adi2]]++;

point[teams[adi1]] = point[teams[adi1]] + 3;

}

else if(goalone<goaltwo)

{

winner[teams[adi2]]++; looser[teams[adi1]]++;

point[teams[adi2]] = point[teams[adi2]] + 3;

}

else

{

draw[teams[adi1]]++; draw[teams[adi2]]++;

point[teams[adi1]] = point[teams[adi1]] + 1; point[teams[adi2]] = point[teams[adi2]] + 1;

}

}

/\*for(it=teams.begin(); it!=teams.end(); ++it)

{

cout<<it->first<<' '<<point[teams[it->first]]<<' '; cout<<total\_khelse[teams[it->first]]<<' '<<winner[teams[it->first]]<<' '; cout<<draw[teams[it->first]]<<' '<<looser[teams[it->first]]<<' '; cout<<team\_goal\_dise[teams[it->first]]<<'

'<<team\_goal\_khaise[teams[it->first]]<<endl;

}\*/

data total\_teams[100]; for(it=teams.begin(); it!=teams.end(); ++it)

{

total\_teams[it->second].point = point[it->second]; total\_teams[it->second].jitse = winner[it→second];

total\_teams[it→second].goal\_diff = team\_goal\_dise[it→second] - team\_goal\_khaise[it→ total\_teams[it->second].goal\_diff = team\_goal\_dise[it->second] - team\_goal\_khaise[it->second];

total\_teams[it->second].goaldise = team\_goal\_dise[it->second]; total\_teams[it->second].khelse = total\_khelse[it->second]; total\_teams[it->second].teamname = it->first;

}

sort(total\_teams, total\_teams+n, cmp); cout<<tourn\_name<<endl;

for(int i=0; i<n; i++)

{

cout<<i+1<<')'<<' '<<total\_teams[i].teamname<<' '<<total\_teams[i].point<<"p, "<<total\_teams[i].khelse<<"g ("<<winner[teams[total\_teams[i].teamname]]<<'-'; cout<<draw[teams[total\_teams[i].teamname]]<<'- '<<looser[teams[total\_teams[i].teamname]]<<"), "<<total\_teams[i].goal\_diff<<"gd (";

cout<<team\_goal\_dise[teams[total\_teams[i].teamname]]<<'- '<<team\_goal\_khaise[teams[total\_teams[i].teamname]]<<')'<<endl;

}

if(test)

printf("\n");

//CLR(total\_teams,0);

}

return 0;

}

# Output:



### Experiment No. 09

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Title:** A) Primary Arithmetic

1. Reverse and Add

**Name:** Shivlal Arun Yadav

**Class**: TE CSE **Batch:** T1 **Roll No:** 3011

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### Primary Arithmetic

**Program:**

#include<bits/stdc++.h> using namespace std;

int main(){

long long a, b, c , \_a, \_b, result, carry;

while (scanf("%lld %lld",&a,&b)){ if ( !a && !b) break;

carry = c = 0; while (a || b){

\_a=a%10; a/=10;

\_b=b% 10; b/=10;

result = \_a + \_b + c; if (result>9){

c=1;

carry++;

}

else

}

c=0;

if (!carry){

printf("No carry operation.\n");

}

else if (carry == 1){

printf("1 carry operation.\n");

}

return 0;

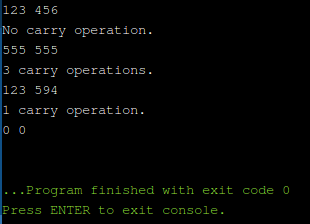
}

}

else

printf("%lld carry operations.\n",carry);

# Output:



## Reverse and Add

**Program:**

#include <iostream> using namespace std; long Reverse(long x)

{

long newX(0); while (x)

{

newX \*= 10; newX += x % 10; x /= 10;

}

return newX;

}

int main()

{

int numberOfCases;

cin >> numberOfCases;

for (int i = 0; i < numberOfCases; ++i)

{

long number; cin >> number;

long reverseNumber = Reverse(number); int count(0);

while (reverseNumber != number)

{

number += reverseNumber;

reverseNumber = Reverse(number);

++count;

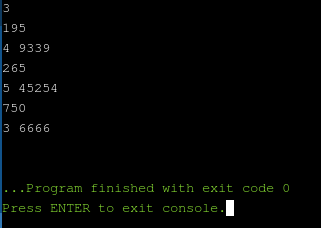
}

cout << count << " " << number << endl;

}

}

# Output:



### Experiment No. 10

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Title:** A) A multiplication game.

B) Polynomial coefficients.

**Name:** Shivlal Arun Yadav

**Class**: TE CSE **Batch:** T1 **Roll No:** 3011

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. **A multiplication game.**

**Program Code:**

#include<iostream> using namespace std; int main(){

long long n; int cont;

while(cin>>n){ if(n==1){

cout<<"Stan wins."<<endl; continue;

}

cont=0; while(n>1){

cont++;

if(cont%2==1) n=(n+8)/9; else n=(n+1)/2;

}

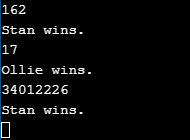
if(cont%2==1) cout<<"Stan wins."<<endl; else cout<<"Ollie wins."<<endl;

}

return 0;

}

## Output:



1. **Polynomial coefficients**.

### Program Code:

#include <iostream>

using namespace std; int main()

{

int fac[13]; fac[0] = 1;

for (int i = 1; i < 13; ++i) fac[i] = i \* fac[i - 1];

int n, k;

while (cin >> n >> k)

{

// Coefficient = n!/(n1! n2! n3!...) int product = 1;

for (int i = 1; i <= k; ++i)

{

int n\_i;

cin >> n\_i;

product \*= fac[n\_i];

}

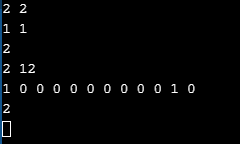
cout << fac[n] / product << endl;

}

return 0;

}

## Output:



### Experiment No. 11

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Title:** A) How many fibs?

* 1. How many pieces of land?

**Name:** Shivlal Arun Yadav

**Class**: TE CSE **Batch:** T1 **Roll No:** 3011

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## How many fibs?

**Program:**

#include<stdio.h> #include<string.h>

#define MAX(x,y) ( (x) >= (y) ? (x) : (y) ) char str1[10000],str2[10000],str3[10000]; char a[10000],b[10000];

bool bigger(char t1[],char t2[]){ int len1=strlen(t1);

int len2=strlen(t2); if(len1>len2)

return true; else if(len1<len2)

return false;

else{

for(int i=0;i<len1;i++)

if(t1[i]>t2[i])

return true; else if(t1[i]<t2[i])

return false;

return true;

}

}

bool contain(char str[]){

if(bigger(str,a)&&bigger(b,str)) return true;

else

}

return false;

void plus(char str1[],char str2[],char str3[]){ int len1=strlen(str1);

int len2=strlen(str2);

int len3=MAX(len1,len2)+1; int i,j,k,temp,carry; str3[len3]='\0';

for(i=len1-1,j=len2-1,k=len3-1,carry=0;i>=0||j>=0;){ if(i>=0&&j>=0)

temp=str1[i--]-'0'+str2[j--]-'0'+carry; else if(i>=0)

temp=str1[i--]-'0'+carry;

else

temp=str2[j--]-'0'+carry;

carry=temp/10; temp%=10;

str3[k--]='0'+temp;

}

str3[k]='0'+carry; if(str3[0]=='0')

memmove(str3,str3+1,sizeof(char)\*len3);

}

int main(){

while(scanf("%s%s",a,b)==2){

if(a[0]=='0'&&b[0]=='0')

break; str1[0]='1',str1[1]='\0';

str2[0]='2',str2[1]='\0';

int count=0; if(contain(str1))

count++; if(contain(str2))

count++; while(!bigger(str2,b)){

plus(str1,str2,str3); if(contain(str3))

count++; memmove(str1,str2,sizeof(str1)); memmove(str2,str3,sizeof(str2));

}

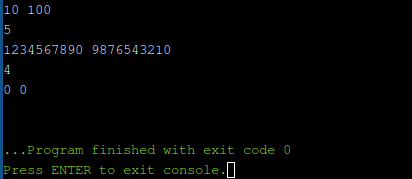
printf("%d\n",count);

}

return 0;

}

## Output:



* + 1. **How many pieces of land?**

**Program:**

import java.io.\*;

import java.math.BigDecimal; import java.math.BigInteger; import java.util.\*;

public class Main

{

public static void main(String args[])

{

BigDecimal a = new BigDecimal("0"); BigDecimal b = new BigDecimal("0"); BigDecimal c = new BigDecimal("0"); Scanner s = new Scanner(System.in); long n;

n = s.nextInt();

for (int i=0; i<n; i++)

{

a = s.nextBigDecimal();

b = a.multiply( a.subtract(BigDecimal.valueOf(1))); b = b.multiply(a.subtract(BigDecimal.valueOf(2))); b = b.multiply(a.subtract(BigDecimal.valueOf(3))); b = b.divide(BigDecimal.valueOf(24));

c = a.multiply(a.subtract(BigDecimal.valueOf(1))); c = c.divide(BigDecimal.valueOf(2));

b = b.add(c);

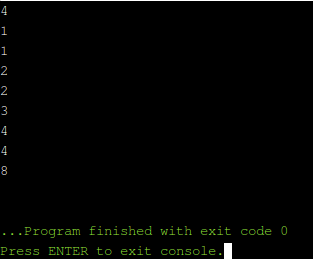
b = b.add(BigDecimal.valueOf(1)); System.out.printf("%s\n", b.toString());

}

}

}

## Output:



### Experiment No. 12

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Title:** A) Complete Tree Labeling

B) Steps

**Name:** Shivlal Arun Yadav

**Class**: TE CSE **Batch:** T1 **Roll No:** 3011

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## Complete Tree Labeling Program:

import java.util.Scanner; import java.util.HashMap; import java.math.BigInteger; class Main {

static int numberOfNodes(int k, int d) {

// number of nodes for a k-ary tree of depth d

return ((int)Math.pow((double)k, (double)d + 1.0) - 1) / (k - 1);

}

static BigInteger factorial(int n) { BigInteger f = BigInteger.ONE; for (int i = 1; i <= n; i++)

f = f.multiply(BigInteger.valueOf(i)); return f;

}

static HashMap<Long, BigInteger> combinationCache = new HashMap<Long, BigInteger>();

static BigInteger combination(int n, int k) { if (k == 0 || n == k)

return BigInteger.ONE; else {

long nk = (long)n << 32 | k;

BigInteger c = combinationCache.get(nk); if (c != null)

return c;

c = factorial(n).divide(factorial(n - k).multiply(factorial(k))); combinationCache.put(nk, c);

return c;

}

}

static HashMap<Long, BigInteger> cache = new HashMap<Long, BigInteger>(); static BigInteger completeTreeLabeling(

int k /\* branching factor \*/, int d /\* depth \*/) { if (k == 1)

return BigInteger.ONE; long kd = (long)k << 32 | d;

BigInteger nrLabeling = cache.get(kd); if (nrLabeling != null)

return nrLabeling;

nrLabeling = factorial(k); if (d > 1) {

// number of nodes for a k-ary tree of depth d int nrNodes = numberOfNodes(k, d);

// number of descendants of the root node for a k-ary tree of depth (d - 1) int nrDescendants = numberOfNodes(k, d - 1) - 1;

// number of labeling for a k-ary tree of depth (d - 1)

BigInteger nrDescendantsLabeling = completeTreeLabeling(k, d - 1); for (int i = nrNodes - 2; i >= nrDescendants; i -= nrDescendants + 1) { BigInteger c = combination(i, nrDescendants);

nrLabeling = nrLabeling.multiply(c).multiply(nrDescendantsLabeling);

}

}

cache.put(kd, nrLabeling); return nrLabeling;

}

public static void main(String[] args) { Scanner sc = new Scanner(System.in); while (sc.hasNextInt()) {

int k = sc.nextInt(); if (sc.hasNextInt()) { int d = sc.nextInt();

System.out.println(completeTreeLabeling(k, d));

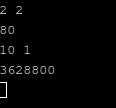
}

}

}

}

## Output:



1. **Steps Program:**

#include <stdio.h>

#include <math.h> int main() {

int t, x, y, L; scanf("%d", &t);

while(t--) {

scanf("%d %d", &x, &y); L = y-x;

int steps = 0, n = (int)sqrt(L); steps = n;

L -= n\*(n+1)/2; while(L > 0) {

while(n\*(n+1)/2 > L)

n--;

if(n\*(n+1)/2 == L)

L = 0, steps += n;

else

}

L -= n, steps ++;

}

return 0;

}

printf("%d\n", steps);

## Output:

