* **WAP to implement set operations (Union, intersection and differences).**

#include<stdio.h>

#include<stdlib.h>

int main()

{

int i,j,k,p,ch,n1,n2,set1[10],set2[10],set3[20],flag;

printf("Enter the size of first set:");

scanf(" %d",&n1);

printf("Enter the elements of first set:");

for(i=0;i<n1;i++)

{

scanf(" %d",&set1[i]);

}

fflush(stdin);

printf("Enter the size of second set:");

scanf(" %d",&n2);

printf("Enter the elements of second set:");

for(i=0;i<n2;i++)

{

scanf(" %d",&set2[i]);

}

while(1)

{

printf("\n1. Union");

printf("\n2. Intersection");

printf("\n3. Difference");

printf("\nEnter your choice(1-3): ");

scanf("%d",&ch);

switch(ch)

{

case 1:

printf("The union is:");

k=0;

for(i=0;i<n1;i++)

{

set3[k]=set1[i];

k++;

}

for(i=0;i<n2;i++)

{

flag=1;

for(j=0;j<n1;j++)

{

if(set2[i]==set1[j])

{

flag=0;

break;

}

}

if(flag==1)

{

set3[k]=set2[i];

k++;

}

}

p=k;

for(k=0;k<p;k++)

{

printf(" %d\t",set3[k]);

}

break;

printf("\n");

case 2:

printf("The intersection is:");

k=0;

for(i=0;i<n2;i++)

{

flag=1;

for(j=0;j<n1;j++)

{

if(set2[i]==set1[j])

{

flag=0;

break;

}

}

if(flag==0)

{

set3[k]=set2[i];

k++;

}

}

p=k;

for(k=0;k<p;k++)

{

printf(" %d\t",set3[k]);

break;

printf("\n");

case 3:

printf("The difference is:");

k=0;

for(i=0;i<n1;i++)

{

flag=1;

for(j=0;j<n2;j++)

{

if(set1[i]==set2[j])

{

flag=0;

break;

}

}

if(flag==1)

{

set3[k]=set1[i];

k++;

}

}

p=k;

for(k=0;k<p;k++)

{

printf(" %d\t",set3[k]);

}

break;

default:

printf("Invalid choice");

exit(0);

}

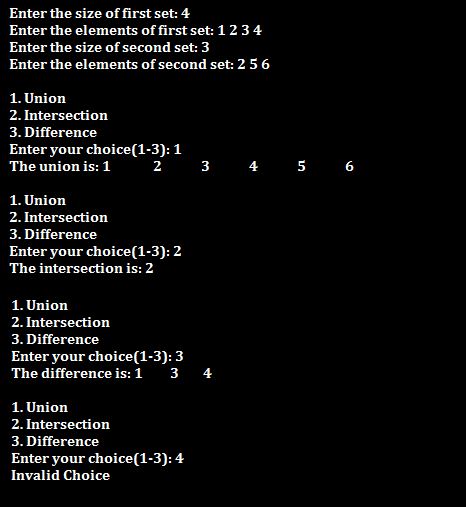
}

}

return 0;

}

**Output:**

****

* **WAP to find the Cartesian product of two sets.**

#include<stdio.h>

#include<conio.h>

int main()

{

int a[10],b[10],c[10],i,j,k,m,n;

printf("Enter the size of first set:");

scanf("%d",&m);

printf("Enter the elements of first set:");

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

{

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

}

printf("Enter the size of second set:");

scanf("%d",&n);

printf("Enter the elements of second set: ");

for(j=0;j<n;j++)

{

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

}

printf("Cartesian product=\n");

printf("{");

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

{

for(j=0;j<n;j++)

{

printf("(%d,%d)",a[i],b[j]);

printf(",");

}

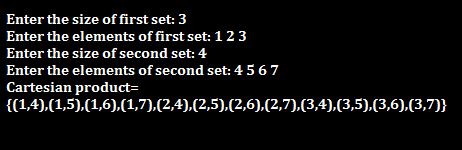
}

printf("}");

return 0;

}

**Output:**

****

* **WAP to implement floor and ceiling function.**

#include <stdio.h>

#include <math.h>

int main()

{

float a;

float f,c;

printf("Enter a float value: ");

scanf("%f",&a);

f=floor(a);

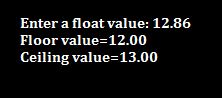
c=ceil(a);

printf("floor value=%.2f \nCeiling value=%.2f",f,c);

return 0;

}

**Output:**

****

* **WAP to find join and meet of the Boolean matrix.**

#include<stdio.h>

#include<conio.h>

int main()

{

int r,m1[10][10],m2[10][10],m[10][10],jo[10][10],i,j;

printf("Enter the number of rows:");

scanf("%d",&r);

printf("Enter the elements of first boolean matrix:");

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

{

for(j=0;j<r;j++)

{

scanf("%d",m1[i][j]);

}

}

printf("Enter the elements of second boolean matrix:");

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

{

for(j=0;j<r;j++)

{

scanf("%d",m2[i][j]);

}

}

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

{

for(j=0;j<r;j++)

{

jo[i][j]=m1[i][j]||m2[i][j];

}

}

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

{

for(j=0;j<r;j++)

{

m[i][j]=m1[i][j]&m2[i][j];

}

}

printf("The joint matrix is:\n");

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

{

for(j=0;j<r;j++)

{

printf("%d",jo[i][j]);

}

printf("\n");

}

printf("The meet matrix is:\n");

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

{

for(j=0;j<r;j++)

{

printf("%d,m[i][j]");

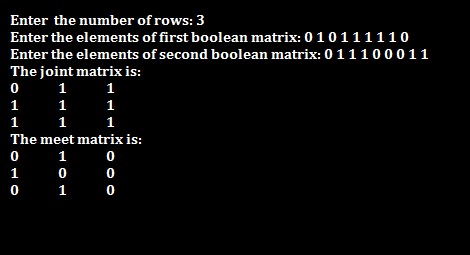
}

printf("\n");

}

}

**Output:**

****

* **WAP to find the product of two Boolean matrices.**

#include<stdio.h>

#include<conio.h>

int main()

{

int m, n, p, q, c, d, k, sum = 0;

int first[10][10], second[10][10], multiply[10][10];

printf("Enter order of first boolean matrix:");

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

printf("Enter elements of first boolean matrix:");

for (c = 0; c < m; c++)

{

for (d = 0; d < n; d++)

{

scanf("%d", &first[c][d]);

}

}

printf("Enter order of second boolean matrix:");

scanf("%d%d", &p, &q);

if (n != p)

{

printf("The matrices can't be multiplied with each other.\n");

}

else

{

printf("Enter elements of second boolean matrix:");

for (c = 0; c < p; c++)

{

for (d = 0; d < q; d++)

{

scanf("%d", &second[c][d]);

}

}

for (c = 0; c < m; c++)

{

for (d = 0; d < q; d++)

{

for (k = 0; k < p; k++)

{

sum = sum || first[c][k]&&second[k][d];

}

multiply[c][d] = sum;

sum = 0;

}

}

printf("Boolean product of the matrices:\n");

for (c = 0; c < m; c++)

{

for (d = 0; d < q; d++)

{

printf("%d\t", multiply[c][d]);

}

printf("\n");

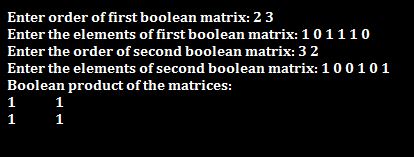
}

}

return 0;

}

**Output:**

****