1.

$(A\times B)\cup(C\times D)=(A\cup C)\times(B\cup D)$

Нехай $(x, y)\in(A\times B)\cup(C\times D)\Leftrightarrow(x,$

$y)\in(A\times B)\ \&\ (x, y)\in(C\times D)\Leftrightarrow$

$(x\in A\ \&\ y\in B)\ \&\ (x\in C\ \&\ y\in D)\Leftrightarrow$

$(x\in A\ \&\ x\in C)\ \&\ (y\in B\ \&\ y\in D)\Leftrightarrow$

$(x\in A\cup C)\ \&\ (y\in B\cup D)\Leftrightarrow(x, y)\in(A\cup C)\times(B\cup D)$.

Вірно.

2.

$R\subset M\times 2^M$

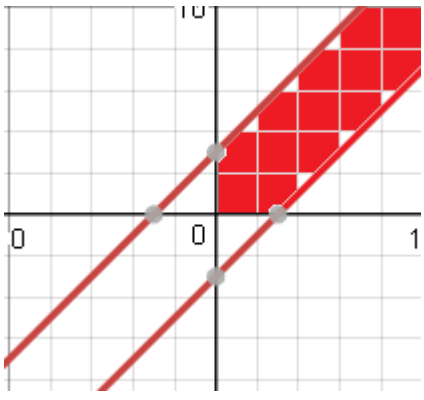
$R\ \ (x, y)\ \ |x\ M\ \&\ x\ \ y\ \&\ \ \ |y|\ x\ \ \ ^{,Де}M\ \ x\ \ \ |x\ Z\ \&\ \ \ |x|\ 1$, *Z – множина цілих чисел.*

	{0}	{-1}	{0}	{1}	{-1;0}	{0;1}	{-1;1}	{-1;0;1}
-1	0	1	0	0	1	0	1	1
0	0	0	1	0	1	1	0	1
1	0	0	0	0	0	1	1	1

3.

$\alpha=\{(x, y)|(x, y)\in R^2\ \&(x-y)^2=9\}$, де R - множина дійсних чисел.

.



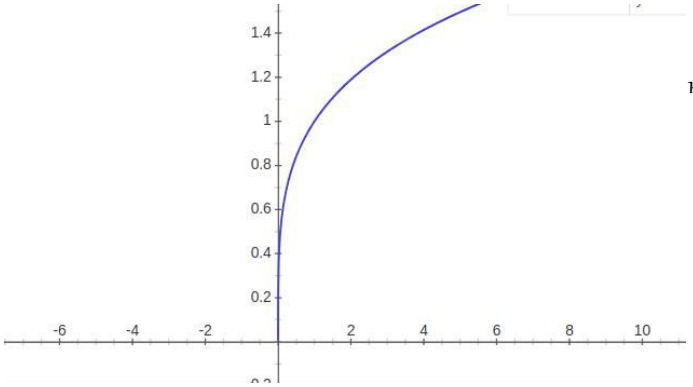
4.

1	1	1	1	1
1	1	1	1	1
1	1	0	1	1
1	1	1	1	1
1	1	1	1	1

* нерефлексивне
* симетричне
* транзитивне

5.

(x, y) (x, y) R^2 & y $\sqrt{x^4}$



При $x>=0$ відношення функціональне і бієктивне, оскільки кожному y відповідає один x та кожному x відповідає один y.

```
#include <stdio.h>
#include <cs50.h>

int main()
{
```

```
int n = 100;
int A[n];
int B[n];

// matrix
int C[n][n];

do
{
   printf("Enter the number of elements in A and B: ");
   n = GetInt();
} while (n < 1);

// initialisation of A and B

for (int i = 0; i < n; i++)
{
   printf("A[%d] = ", i + 1);
   A[i] = GetInt();
```

```
}
for (int i = 0; i < n; i++)
{
    printf("B[%d] = ", i + 1);
    B[i] = GetInt();
}

// matrix

for (int i = 0; i < n; i++)
{
    printf("| ");
    for (int j = 0; j < n; j++)
    {
        if (2*A[i]-B[j]  < 3)
        {
            C[i][j] = 1;
            printf("%d ", C[i][j]);
        }
```

```
        else
        {
            C[i][j] = 0;
            printf("%d ", C[i][j]);
        }
    }
    printf("|");
    printf("\n");
}

// reflexivity

for (int j = 0; j < n; j++)
{
    if (C[j][j] != 1)
    {
        printf("Not reflexive");
        break;
    }
```

```
    else if (j == n - 1)
        printf("Reflexive");
}
printf("\n");

// antireflexivity

for (int j = 0; j < n; j++)
{
    if (C[j][j] != 0)
    {
        printf("Not antireflexive");
        break;
    }
    else if (j == n - 1)
        printf("Antireflexive");
}
printf("\n");
```

```
// symmetry


for (int i = 0, k = 0; i < n; i++)
{
   for (int j = 0; j < n; j++)
   {
      if (C[i][j] != C[j][i] && i != j)
      {
         k = 1;
         printf("Not symmetic");
         break;
      }
      else if (i == n - 1 && j == n - 1)
         printf("Symmetric");
   }
   if (k)
      break;
}
printf("\n");
```

```
// antisymmetry


for (int i = 0, l = 0; i < n; i++)
{
   for (int j = 0; j < n; j++)
   {
      if (C[i][j] == C[j][i] && i != j)
      {
         l = 1;
         printf("Not antisymmetic");
         break;
      }
      else if (i == n - 1 && j == n - 1)
         printf("Antisymmetric");
   }
   if (l)
      break;
}
```

```
printf("\n");


// transitivity


bool transitive = true;
bool antitransitive = true;
for (int i = 0; i < n; i++)
{
   for (int j = 0; j < n; j++)
   {
      for (int k = 0; k < n; k++)
      {
         if (j == k == i)
            continue;
         transitive = transitive & (!C[i][j] || C[j][k] || C[i][k]);
         antitransitive = antitransitive & (!C[i][j] || !C[j][k] || !C[i][k]);
      }
   }
}
```

```
    if (transitive)
        printf("Transitive\n");
    else
        printf("Not transitive\n");

    if (antitransitive)
        printf("Antitransitive\n");
    else
        printf("Not antitransitive\n");

    return 0;
}
```