2.1 (a) – (g).

(a) "isBrotherOf" on the set of people.

Symmetric: False, sister can't be brother of

Reflexive: False, can't be a brother to yourself

Transitive: True, 1 is brother of 2 and 2 is brother of 3 so 1 is brother of 3

Anti-symmetric: False, the two are not the same person

(b) "isFatherOf" on the set of people

Symmetric: False, can't be a father to self

Reflexive: False, child can't be father of Father

Transitive: False, 2 generations back is grand father

Anti-symmetric: True, because father can't go both ways

(c) The relation $R = \{h_x, y_i \mid x^2 + y^2 = 1\}$ for real numbers x and y.

Symmetric: False, 1 is the only one in the relation, (2,2) = 8 not 1

Reflexive: False, 1 is the only one in the relation, (2,2) = 8 not 1

Transitive: False, (.4 , .916) -> (.916 ,.4). The .4 isn't equal to .4

Anti-symmetric: False, (.5 , .707) is not the same as (.707 , .5)

(d) The relation $R = \{h_x, h_y \mid x^2 = y^2\}$ for real numbers x and y.

Symmetric: true, $1^2 = 1^2$

Reflexive: True, a = b and b = a

Transitive: True, $1^2 = 2^2 \wedge 2^2 = 3^2 \rightarrow 1^2 = 3^2$

Anti-symmetric: False because $1^2 \, ! = 2^2$

(e) The relation $R = \{h_x, h_y \mid x \bmod y = 0\}$ for $x, y \in \{1,2,3,4\}$

Symmetric: False, 1 mod 2 != 0

Reflexive: True, 1 mod 1 is zero and is same for the rest of the set

Transitive: True, they all equal 0

Anti-symmetric: True,1 mod 2 != 0

(f) The empty relation∅(i.e., the relation with no ordered pairs for which

Symmetric: True, you don't have (a,b) so you don't need (b,a)

Reflexive: False, everything isn't related to itself

Transitive: True, you don't have (a,b) so don’t need (b,c) for (a,c)

Anti-symmetric: True, its not reflexive

2.2 (a) – (f).

For each of the following relations, either prove that it is an equivalence relation or prove that it is not an equivalence relation.

(a) For integers a and b, a ≡ b if and only if a + b is even.

This is not an equivalence relation because 1+2 is an odd number

(b) For integers a and b, a ≡ b if and only if a + b is odd.

Is not an equivalence relation because reflexive case fails 1+1 = 2 <--even

(c) For nonzero rational numbers a and b, a ≡ b if and only if a×b > 0.

This is an equivalence relation because its reflexive(1,1)>0, symmetric(1,2)(2,1)>0, and transitive(1,2)(2,3)(1,3)>0

(d) For nonzero rational numbers a and b, a ≡ b if and only if a/b is an integer.

Is not an equivalence relation because symmetric fails (b,a) is fraction

(e) For rational numbers a and b, a ≡ b if and only if a−b is an integer.

This is an equivalence relation because its reflexive(1-1=0 , symmetric(1-2=-1,2-1=1), and transitive(1-2=-1 , 2-3=-1 , 1-3=-2)

(f) For rational numbers a and b, a ≡ b if and only if|a−b|≤ 2.

Is not an equivalence relation because it is not transitive

2.3 (a) – (f).

State whether each of the following relations is a partial ordering, and explain why or why not.

(a) “isFatherOf” on the set of people.

Is not a partial order because a != c for transitive

(b) “isAncestorOf” on the set of people.

Is a partial order because antisymmetric case the (b,a) isnt there and (a,b)(b,c)->(a,c)

(c) “isOlderThan” on the set of people.

Is a partial order because antisymmetric case the (b,a) isnt there and (a,b)(b,c)->(a,c)

(d) “isSisterOf” on the set of people.

Is not a partial order because a != c for transitive

(e) {ha,bi,ha,ai,hb,ai}on the set{a,b}.

not a partial order because its not transitive

(f) {h2,1i,h1,3i,h2,3i}on the set{1,2,3}.

not a partial order because its not transitive

2.15. Your function should have the prototype void perms(int A[], int n); Test your function separately and include in your answer what the output would be if you called your function with arguments int A[4] = {1,2,3,4}, n = 4.

Write a function to print all of the permutations for the elements of an array containing n distinct integer values

```
#include<iostream>
using namespace std;

void permute(int n, int *OtherNumbers = NULL, int size = 0){
    for(int i = 0; i < n; i++){
        bool taken = false;
        for (int j = 0; j < size; ++j){
            if (i == OtherNumbers[j]){
                taken = true;
            }
        }

        if(!taken){
            int *copyList = new int[size+1];
            for (int k = 0; k < size; k++){
                copyList[k] = OtherNumbers[k];
            }
            copyList[size] = i;

            if ( size+1 == n){
                for (int l = 0; l < size+1; l++){
                    cout << copyList[l] << ' ';
                }
                cout << endl;
            }
            else{
                permute(n,copyList,size+1);
            }
            delete[] copyList;
        }
    }
}

int main()
{
    int n;
    cout << "Input a positive integer: ";
    cin >> n;
    cout << endl << endl;
    permute(n);

    system("PAUSE");
    return 0;
}
```

```
Input a positive integer: 4


0 1 2 3
0 1 3 2
0 2 1 3
0 2 3 1
0 3 1 2
0 3 2 1
1 0 2 3
1 0 3 2
1 2 0 3
1 2 3 0
1 3 0 2
1 3 2 0
2 0 1 3
2 0 3 1
2 1 0 3
2 1 3 0
2 3 0 1
2 3 1 0
3 0 1 2
3 0 2 1
3 1 0 2
3 1 2 0
3 2 0 1
3 2 1 0
Press any key to continue . . .
```

2.16. Your function should have the prototype void subsets(int n); Subsets should be printed, for example, as {}, {1}, {1,2}, etc.

Write a recursive algorithm to print all of the subsets for the set of the first n positive integers

```
#include<iostream>
using namespace std;

//initialize
void subset(int n) {
    int arr[10];
    for (int i = 1; i <= n; i++) {
        arr[i] = i;
    }
    printset(arr, 1, 4);
}

/*This is all Psuedo Code
void printset(int sets[], int l, int h) {
    if (l > h) {
        int i = 0;
        for (all entries i; in bool sets[] which are true)
            cout << sets[i];
    }
    else {
        set sets[l] to true <--going towards the left side
        recursively call printset(sets,l+1,h)
        set sets[l] to false <--going towards right side
        recursively call printset(sets,l+1,h)
    }
}*/

int main() {
    int num;
    cout << "Enter number:";
    cin >> num;
    cout << endl;
    subset(num);

    system("PAUSE");
    return 0;
}
```

2.22 – **2.24**.

22) Prove Equation 2.2 using mathematical induction.

2.2) $\sum_{i=1}^{n} i^2 = \frac{2n^3+3n^2+n}{6} = \frac{n(2n+1)(n+1)}{6}$

Left side = $1^2 = 1$ | right side = $\frac{1(1+1)(2*1+1)}{6} = 1$

Both sides are equal so p(1) is true

Now p(n)

$1^2 + 2^2 + 3^2 + ... + n^2 = \frac{n(n+1)(2n+1)}{6}$ now, p(n+1)

$1^2 + 2^2 + 3^2 + ... + n^2 + (n + 1)^2 = \frac{n(n+1)(2n+1)}{6} + (n+1)^2$

$= \frac{(n+1)[n(2n+1)+6(n+1)]}{6}$

$= \frac{(n+1)[2n^2+7n+6]}{6}$

$= \frac{(n+1)[(n+2)(2n+3)]}{6}$ -> which is the same as the statement for p(n+1)

23) Prove Equation 2.6 using mathematical induction.

2.6) $\sum_{i=1}^{n} \frac{1}{2^i} = 1 - \frac{1}{2^n}$

24) Prove Equation 2.7 using mathematical induction.

2.7) $\sum_{i=0}^{n} 2^i = 2^{n+1} - 1$

2.28.

Prove that $Fib(n) < \left(\frac{5}{3}\right)^n$.

Base cases: n=1: Fib(1) = 1, and $\left(\frac{5}{3}\right)^1 = \frac{5}{3} > 1$

n=2: Fib(2) =1 < $\left(\frac{5}{3}\right)^2$.

Assume $Fib(m) < \left(\frac{5}{3}\right)^m$ $\forall\, m < n$, where n > 2.

$Fib(n-1) + Fib\, n(n-2) < \left(\frac{5}{3}\right)^{n-1} + \left(\frac{5}{3}\right)^{n-2}$

$= \left(\frac{5}{3}\right)^{n-2} (\frac{5}{3} + 1)$

$= \left(\frac{5}{3}\right)^{n-2} \frac{8}{3} < \left(\frac{5}{3}\right)^{n}$

Since $\frac{8}{3} < \frac{25}{9} <=> \left(\frac{5}{3}\right)^{2}$