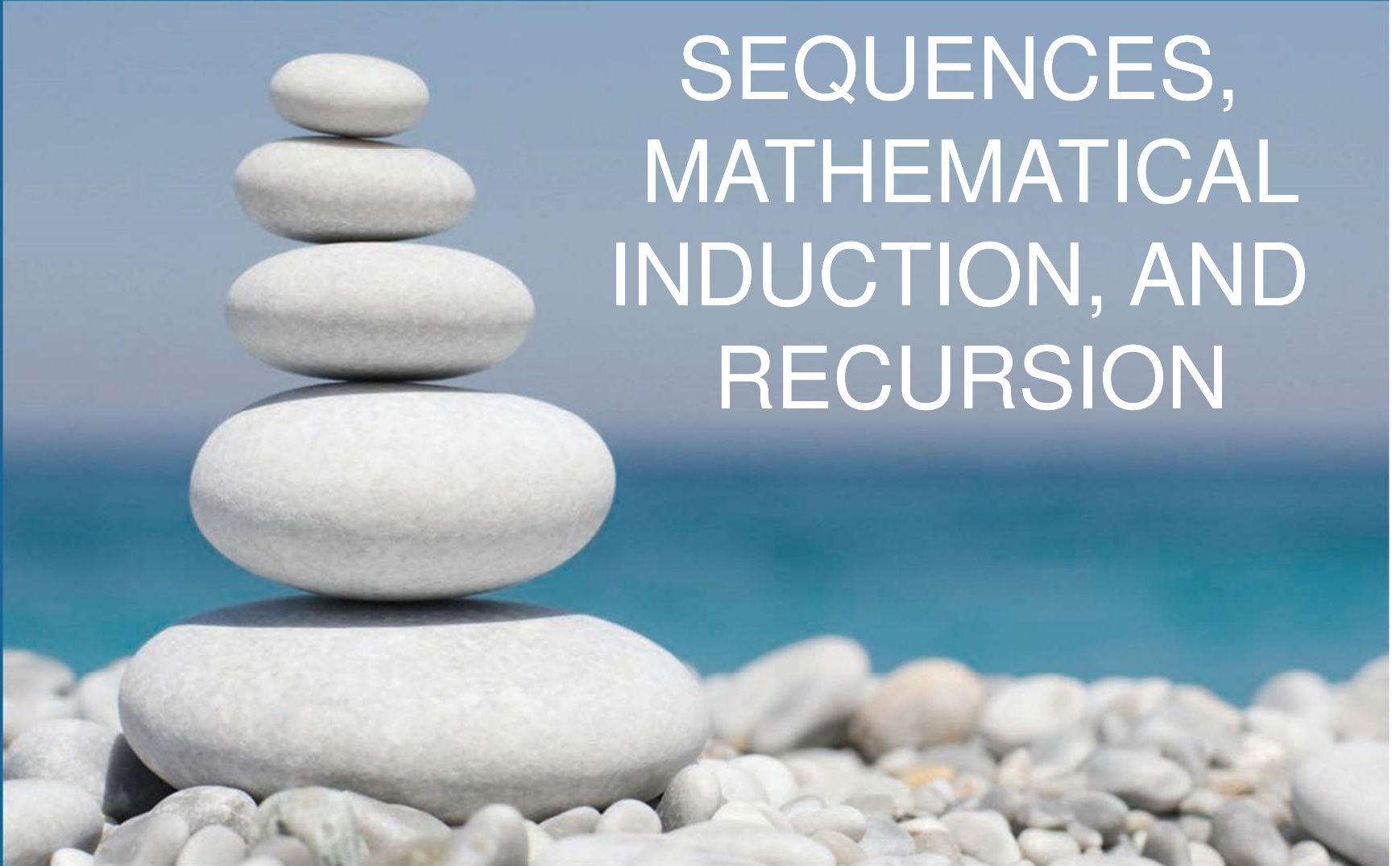


CHAPTER 5

SEQUENCES, MATHEMATICAL INDUCTION, AND RECURSION



SECTION 5.1

Sequences



Sequences

Imagine that a person decides to count his ancestors. He has two parents, four grandparents, eight great-grandparents, and so forth. These numbers can be written in a row as

2, 4, 8, 16, 32, 64, 128,...

The symbol “...” is called an *ellipsis*. It is shorthand for “and so forth.”

To express the pattern of the numbers, suppose that each is labeled by an integer giving its position in the row.

Position in the row	1	2	3	4	5	6	7...
Number of ancestors	2	4	8	16	32	64	128...



Sequences

The number corresponding to position 1 is 2, which equals 2^1 . The number corresponding to position 2 is 4, which equals 2^2 .

For positions 3, 4, 5, 6, and 7, the corresponding numbers are 8, 16, 32, 64, and 128, which equal 2^3 , 2^4 , 2^5 , 2^6 , and 2^7 , respectively.

For a general value of k , let A_k be the number of ancestors in the k th generation back. The pattern of computed values strongly suggests the following for each k :

$$A_k = 2^k.$$



Sequences

- **Definition**

A **sequence** is a function whose domain is either all the integers between two given integers or all the integers greater than or equal to a given integer.

We typically represent a sequence as a set of elements written in a row. In the sequence denoted

$$a_m, a_{m+1}, a_{m+2}, \dots, a_n,$$

each individual element a_k (read “ a sub k ”) is called a **term**.



Sequences

The k in a_k is called a **subscript** or **index**, m (which may be any integer) is the subscript of the **initial term**, and n (which must be greater than or equal to m) is the subscript of the **final term**. The notation

$$a_m, a_{m+1}, a_{m+2}, \dots$$

denotes an **infinite sequence**. An **explicit formula** or **general formula** for a sequence is a rule that shows how the values of a_k depend on k .

The following example shows that it is possible for two different formulas to give sequences with the same terms.



Example 1 – Finding Terms of Sequences Given by Explicit Formulas

Define sequences a_1, a_2, a_3, \dots and b_2, b_3, b_4, \dots by the following explicit formulas:


$$a_k = \frac{k}{k+1} \quad \text{for all integers } k \geq 1,$$

$$b_i = \frac{i-1}{i} \quad \text{for all integers } i \geq 2.$$

Compute the first five terms of both sequences.

Solution:

$$a_1 = \frac{1}{1+1} = \frac{1}{2} \qquad b_2 = \frac{2-1}{2} = \frac{1}{2}$$



Example 1 – *Solution*

cont'd

$$a_2 = \frac{2}{2+1} = \frac{2}{3}$$

$$b_3 = \frac{3-1}{3} = \frac{2}{3}$$

$$a_3 = \frac{3}{3+1} = \frac{3}{4}$$

$$b_4 = \frac{4-1}{4} = \frac{3}{4}$$

$$a_4 = \frac{4}{4+1} = \frac{4}{5}$$

$$b_5 = \frac{5-1}{5} = \frac{4}{5}$$

$$a_5 = \frac{5}{5+1} = \frac{5}{6}$$

$$b_6 = \frac{6-1}{6} = \frac{5}{6}$$

As you can see, the first terms of both sequences are $\frac{1}{2}, \frac{2}{3}, \frac{3}{4}, \frac{4}{5}, \frac{5}{6}$; in fact, it can be shown that all terms of both sequences are identical.



Summation Notation



Summation Notation

Consider again the example in which $A_k = 2^k$ represents the number of ancestors a person has in the k th generation back. What is the total number of ancestors for the past six generations?

The answer is

$$A_1 + A_2 + A_3 + A_4 + A_5 + A_6 = 2^1 + 2^2 + 2^3 + 2^4 + 2^5 + 2^6 = 126.$$

It is convenient to use a shorthand notation to write such sums.



Summation Notation

In 1772 the French mathematician Joseph Louis Lagrange introduced the capital Greek letter sigma, Σ , to denote the word *sum* (or *summation*), and defined the summation notation as follows:

• Definition

If m and n are integers and $m \leq n$, the symbol $\sum_{k=m}^n a_k$, read the **summation from k equals m to n of a -sub- k** , is the sum of all the terms $a_m, a_{m+1}, a_{m+2}, \dots, a_n$. We say that $a_m + a_{m+1} + a_{m+2} + \dots + a_n$ is the **expanded form** of the sum, and we write

$$\sum_{k=m}^n a_k = a_m + a_{m+1} + a_{m+2} + \dots + a_n.$$

We call k the **index** of the summation, m the **lower limit** of the summation, and n the **upper limit** of the summation.



Example 4 – *Computing Summations*

Let $a_1 = -2$, $a_2 = -1$, $a_3 = 0$, $a_4 = 1$, and $a_5 = 2$. Compute the following:


a. $\sum_{k=1}^5 a_k$

b. $\sum_{k=2}^2 a_k$

c. $\sum_{k=1}^2 a_{2k}$

Solution:

a.
$$\begin{aligned}\sum_{k=1}^5 a_k &= a_1 + a_2 + a_3 + a_4 + a_5 \\ &= (-2) + (-1) + 0 + 1 + 2 \\ &= 0\end{aligned}$$



Example 4 – *Solution*

cont'd

$$\mathbf{b.} \sum_{k=2}^2 a_k = a_2$$

$$= -1$$

$$\mathbf{c.} \sum_{k=1}^2 a_{2k} = a_{2 \cdot 1} + a_{2 \cdot 2}$$

$$= a_2 + a_4$$

$$= -1 + 1$$

$$= 0$$



Summation Notation

Oftentimes, the terms of a summation are expressed using an explicit formula.

For instance, it is common to see summations such as

$$\sum_{k=1}^5 k^2 \quad \text{or} \quad \sum_{i=0}^8 \frac{(-1)^i}{i+1}.$$



Example 6 – Changing from Summation Notation to Expanded Form

Write the following summation in expanded form:

$$\sum_{i=0}^n \frac{(-1)^i}{i+1}.$$

Solution:

$$\sum_{i=0}^n \frac{(-1)^i}{i+1} = \frac{(-1)^0}{0+1} + \frac{(-1)^1}{1+1} + \frac{(-1)^2}{2+1} + \frac{(-1)^3}{3+1} + \dots + \frac{(-1)^n}{n+1}$$

$$= \frac{1}{1} + \frac{(-1)}{2} + \frac{1}{3} + \frac{(-1)}{4} + \dots + \frac{(-1)^n}{n+1}$$

$$= 1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + \dots + \frac{(-1)^n}{n+1}$$



Example 7 – Changing from Expanded Form to Summation Notation

Express the following using summation notation:

$$\frac{1}{n} + \frac{2}{n+1} + \frac{3}{n+2} + \cdots + \frac{n+1}{2n}.$$

Solution:

The general term of this summation can be expressed as $\frac{k+1}{n+k}$ for integers k from 0 to n .

Hence

$$\frac{1}{n} + \frac{2}{n+1} + \frac{3}{n+2} + \cdots + \frac{n+1}{2n} = \sum_{k=0}^n \frac{k+1}{n+k}.$$



Summation Notation

A more mathematically precise definition of summation, called a *recursive definition*, is the following:

If m is any integer, then

$$\sum_{k=m}^m a_k = a_m \quad \text{and} \quad \sum_{k=m}^n a_k = \sum_{k=m}^{n-1} a_k + a_n \quad \text{for all integers } n > m.$$

When solving problems, it is often useful to rewrite a summation using the recursive form of the definition, either by separating off the final term of a summation or by adding a final term to a summation.



Example 9 – Separating Off a Final Term and Adding On a Final Term

a. Rewrite $\sum_{i=1}^{n+1} \frac{1}{i^2}$ by separating off the final term.

b. Write $\sum_{k=0}^n 2^k + 2^{n+1}$ as a single summation.

Solution:

a.
$$\sum_{i=1}^{n+1} \frac{1}{i^2} = \sum_{i=1}^n \frac{1}{i^2} + \frac{1}{(n+1)^2}$$

b.
$$\sum_{k=0}^n 2^k + 2^{n+1} = \sum_{k=0}^{n+1} 2^k$$



Summation Notation

In certain sums each term is a difference of two quantities.

When you write such sums in expanded form, you sometimes see that all the terms cancel except the first and the last.

Successive cancellation of terms collapses the sum like a telescope.



Example 10 – *A Telescoping Sum*

Some sums can be transformed into telescoping sums, which then can be rewritten as a simple expression.

For instance, observe that

$$\frac{1}{k} - \frac{1}{k+1} = \frac{(k+1) - k}{k(k+1)} = \frac{1}{k(k+1)}.$$

Use this identity to find a simple expression for $\sum_{k=1}^n \frac{1}{k(k+1)}$.



Example 10 – *Solution*

$$\sum_{k=1}^n \frac{1}{k(k+1)} = \sum_{k=1}^n \left(\frac{1}{k} - \frac{1}{k+1} \right)$$

$$= \left(\frac{1}{1} - \frac{1}{2} \right) + \left(\frac{1}{2} - \frac{1}{3} \right) + \left(\frac{1}{3} - \frac{1}{4} \right) + \cdots + \left(\frac{1}{n-1} - \frac{1}{n} \right) + \left(\frac{1}{n} - \frac{1}{n+1} \right)$$

$$= 1 - \frac{1}{n+1}.$$



Product Notation



Product Notation

The notation for the product of a sequence of numbers is analogous to the notation for their sum. The Greek capital letter pi, Π , denotes a product. For example,

$$\prod_{k=1}^5 a_k = a_1 a_2 a_3 a_4 a_5.$$

• Definition

If m and n are integers and $m \leq n$, the symbol $\prod_{k=m}^n a_k$, read the **product from k equals m to n of a -sub- k** , is the product of all the terms $a_m, a_{m+1}, a_{m+2}, \dots, a_n$.

We write

$$\prod_{k=m}^n a_k = a_m \cdot a_{m+1} \cdot a_{m+2} \cdots a_n.$$



Product Notation

A recursive definition for the product notation is the following: If m is any integer, then

$$\prod_{k=m}^m a_k = a_m \quad \text{and} \quad \prod_{k=m}^n a_k = \left(\prod_{k=m}^{n-1} a_k \right) \cdot a_n \quad \text{for all integers } n > m.$$



Example 11 – *Computing Products*

Compute the following products:

a. $\prod_{k=1}^5 k$

b. $\prod_{k=1}^1 \frac{k}{k+1}$

Solution:

a. $\prod_{k=1}^5 k = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 = 120$

b. $\prod_{k=1}^1 \frac{k}{k+1} = \frac{1}{1+1} = \frac{1}{2}$



Properties of Summations and Products



Properties of Summations and Products

The following theorem states general properties of summations and products.

Theorem 5.1.1

If $a_m, a_{m+1}, a_{m+2}, \dots$ and $b_m, b_{m+1}, b_{m+2}, \dots$ are sequences of real numbers and c is any real number, then the following equations hold for any integer $n \geq m$:

$$1. \sum_{k=m}^n a_k + \sum_{k=m}^n b_k = \sum_{k=m}^n (a_k + b_k)$$

$$2. c \cdot \sum_{k=m}^n a_k = \sum_{k=m}^n c \cdot a_k \quad \text{generalized distributive law}$$

$$3. \left(\prod_{k=m}^n a_k \right) \cdot \left(\prod_{k=m}^n b_k \right) = \prod_{k=m}^n (a_k \cdot b_k).$$



Example 12 – Using Properties of Summation and Product

Let $a_k = k + 1$ and $b_k = k - 1$ for all integers k . Write each of the following expressions as a single summation or product:

a. $\sum_{k=m}^n a_k + 2 \cdot \sum_{k=m}^n b_k$ **b.** $\left(\prod_{k=m}^n a_k \right) \cdot \left(\prod_{k=m}^n b_k \right)$

Solution:

a. $\sum_{k=m}^n a_k + 2 \cdot \sum_{k=m}^n b_k = \sum_{k=m}^n (k + 1) + 2 \cdot \sum_{k=m}^n (k - 1)$ by substitution

$= \sum_{k=m}^n (k + 1) + \sum_{k=m}^n 2 \cdot (k - 1)$ by Theorem 5.1.1 (2)



Example 12 – Using Properties of Summation and Product

$$= \sum_{k=m}^n ((k+1) + 2 \cdot (k-1)) \quad \text{by Theorem 5.1.1 (1)}$$

$$= \sum_{k=m}^n (3k - 1) \quad \text{by algebraic simplification}$$

b. $\left(\prod_{k=m}^n a_k \right) \cdot \left(\prod_{k=m}^n b_k \right) = \left(\prod_{k=m}^n (k+1) \right) \cdot \left(\prod_{k=m}^n (k-1) \right)$ by substitution

$$= \prod_{k=m}^n (k+1) \cdot (k-1) \quad \text{by Theorem 5.1.1 (3)}$$

$$= \prod_{k=m}^n (k^2 - 1) \quad \text{by algebraic simplification}$$



Change of Variable



Change of Variable

Observe that $\sum_{k=1}^3 k^2 = 1^2 + 2^2 + 3^2$

and also that $\sum_{i=1}^3 i^2 = 1^2 + 2^2 + 3^2$.

Hence $\sum_{k=1}^3 k^2 = \sum_{i=1}^3 i^2$.

This equation illustrates the fact that the symbol used to represent the index of a summation can be replaced by any other symbol as long as the replacement is made in each location where the symbol occurs.



Change of Variable

As a consequence, the index of a summation is called a dummy variable.

A **dummy variable** is a symbol that derives its entire meaning from its local context. Outside of that context (both before and after), the symbol may have another meaning entirely.

A general procedure to transform the first summation into the second is illustrated in Example 13.



Example 13 – *Transforming a Sum by a Change of Variable*

Transform the following summation by making the specified change of variable.

$$\text{summation: } \sum_{k=0}^6 \frac{1}{k+1} \quad \text{change of variable: } j = k + 1$$

Solution:

First calculate the lower and upper limits of the new summation:

$$\text{When } k = 0, \quad j = k + 1 = 0 + 1 = 1.$$

$$\text{When } k = 6, \quad j = k + 1 = 6 + 1 = 7.$$

Thus the new sum goes from $j = 1$ to $j = 7$.



Example 13 – *Solution*

cont'd

Next calculate the general term of the new summation. You will need to replace each occurrence of k by an expression in j :

Since $j = k + 1$, then $k = j - 1$.

$$\text{Hence } \frac{1}{k + 1} = \frac{1}{(j - 1) + 1} = \frac{1}{j}.$$

Finally, put the steps together to obtain

$$\sum_{k=0}^6 \frac{1}{k + 1} = \sum_{j=1}^7 \frac{1}{j}.$$



Change of Variable

Sometimes it is necessary to shift the limits of one summation in order to add it to another.

A general procedure for making such a shift when the upper limit is part of the summand is illustrated in the next example.



Example 14 – *When the Upper Limit Appears in the Expression to Be Summed*

- a.** Transform the following summation by making the specified change of variable.

$$\text{summation: } \sum_{k=1}^{n+1} \left(\frac{k}{n+k} \right) \text{ change of variable: } j = k - 1$$

- b.** Transform the summation obtained in part (a) by changing all j 's to k 's.



Example 14 – *Solution*

a. When $k = 1$, then $j = k - 1 = 1 - 1 = 0$. (So the new lower limit is 0.)

When $k = n + 1$, then $j = k - 1 = (n + 1) - 1 = n$. (So the new upper limit is n .)

Since $j = k - 1$, then $k = j + 1$. Also note that n is a constant as far as the terms of the sum are concerned.

It follows that

$$\frac{k}{n + k} = \frac{j + 1}{n + (j + 1)}$$

and so the general term of the new summation is

$$\frac{j + 1}{n + (j + 1)}.$$



Example 14 – *Solution*

cont'd

Therefore,

$$\sum_{k=1}^{n+1} \frac{k}{n+k} = \sum_{j=0}^n \frac{j+1}{n+(j+1)}. \quad 5.1.3$$

b. Changing all the j 's to k 's in the right-hand side of equation (5.1.3) gives

$$\sum_{j=0}^n \frac{j+1}{n+(j+1)} = \sum_{k=0}^n \frac{k+1}{n+(k+1)} \quad 5.1.4$$

Combining equations (5.1.3) and (5.1.4) results in

$$\sum_{k=1}^{n+1} \frac{k}{n+k} = \sum_{k=0}^n \frac{k+1}{n+(k+1)}.$$



Factorial and “ n Choose r ” Notation



Factorial and “ n Choose r ” Notation

The product of all consecutive integers up to a given integer occurs so often in mathematics that it is given a special notation—*factorial* notation.

• Definition

For each positive integer n , the quantity **n factorial** denoted $n!$, is defined to be the product of all the integers from 1 to n :

$$n! = n \cdot (n - 1) \cdots 3 \cdot 2 \cdot 1.$$

Zero factorial, denoted $0!$, is defined to be 1:

$$0! = 1.$$



Factorial and “ n Choose r ” Notation

A recursive definition for factorial is the following: Given any nonnegative integer n ,

$$n! = \begin{cases} 1 & \text{if } n = 0 \\ n \cdot (n - 1)! & \text{if } n \geq 1. \end{cases}$$

The next example illustrates the usefulness of the recursive definition for making computations.



Example 16 – Computing with Factorials

Simplify the following expressions:

a. $\frac{8!}{7!}$

b. $\frac{5!}{2! \cdot 3!}$

c. $\frac{1}{2! \cdot 4!} + \frac{1}{3! \cdot 3!}$

d. $\frac{(n+1)!}{n!}$

e. $\frac{n!}{(n-3)!}$

Solution:

a. $\frac{8!}{7!} = \frac{8 \cdot \cancel{7!}}{\cancel{7!}} = 8$

b. $\frac{5!}{2! \cdot 3!} = \frac{5 \cdot 4 \cdot \cancel{3!}}{2! \cdot \cancel{3!}} = \frac{5 \cdot 4}{2 \cdot 1} = 10$



Example 16 – *Solution*

cont'd

$$\mathbf{C.} \quad \frac{1}{2! \cdot 4!} + \frac{1}{3! \cdot 3!} = \frac{1}{2! \cdot 4!} \cdot \frac{3}{3} + \frac{1}{3! \cdot 3!} \cdot \frac{4}{4}$$

by multiplying each numerator and denominator by just what is necessary to obtain a common denominator

$$= \frac{3}{3 \cdot 2! \cdot 4!} + \frac{4}{3! \cdot 4 \cdot 3!}$$

by rearranging factors

$$= \frac{3}{3! \cdot 4!} + \frac{4}{3! \cdot 4!}$$

because $3 \cdot 2! = 3!$ and $4 \cdot 3! = 4!$

$$= \frac{7}{3! \cdot 4!}$$

by the rule for adding fractions with a common denominator

$$= \frac{7}{144}$$



Example 16 – *Solution*

cont'd

$$\begin{aligned} \mathbf{d.} \quad \frac{(n+1)!}{n!} &= \frac{(n+1) \cdot \cancel{n!}}{\cancel{n!}} \\ &= n+1 \end{aligned}$$

$$\begin{aligned} \mathbf{e.} \quad \frac{n!}{(n-3)!} &= \frac{n \cdot (n-1) \cdot (n-2) \cdot \cancel{(n-3)!}}{\cancel{(n-3)!}} \\ &= n \cdot (n-1) \cdot (n-2) \\ &= n^3 - 3n^2 + 2n \end{aligned}$$



Factorial and “ n Choose r ” Notation

An important use for the factorial notation is in calculating values of quantities, called n choose r , that occur in many branches of mathematics, especially those connected with the study of counting techniques and probability.

• Definition

Let n and r be integers with $0 \leq r \leq n$. The symbol

$$\binom{n}{r}$$

is read “ n choose r ” and represents the number of subsets of size r that can be chosen from a set with n elements.

Observe that the definition implies that $\binom{n}{r}$ will always be an integer because it is a number of subsets.



Factorial and “ n Choose r ” Notation

The computational formula:

- Formula for Computing $\binom{n}{r}$

For all integers n and r with $0 \leq r \leq n$,

$$\binom{n}{r} = \frac{n!}{r!(n-r)!}.$$

Many electronic calculators have keys for computing values of $\binom{n}{r}$. These are denoted in various ways such as nCr , $C(n, r)$, nC_r , and $C_{n,r}$.

The letter C is used because the quantities $\binom{n}{r}$ are also called *combinations*. Sometimes they are referred to as *binomial coefficients* because of the connection with the binomial theorem.



Example 17 – Computing $\binom{n}{r}$ by Hand

Use the formula for computing $\binom{n}{r}$ to evaluate the following expressions:

a. $\binom{8}{5}$ b. $\binom{4}{0}$ c. $\binom{n+1}{n}$

Solution:

$$\begin{aligned} \text{a. } \binom{8}{5} &= \frac{8!}{5!(8-5)!} \\ &= \frac{8 \cdot 7 \cdot \cancel{6} \cdot \cancel{5} \cdot \cancel{4} \cdot \cancel{3} \cdot 2 \cdot 1}{(\cancel{5} \cdot \cancel{4} \cdot \cancel{3} \cdot \cancel{2} \cdot 1) \cdot (\cancel{3} \cdot 2 \cdot 1)} \\ &= 56. \end{aligned}$$

always cancel common factors
before multiplying

Example 17 – Solution

cont'd

$$\begin{aligned}\mathbf{b.} \binom{4}{4} &= \frac{4!}{4!(4-4)!} \\ &= \frac{4!}{4!0!} \\ &= \frac{\cancel{4 \cdot 3 \cdot 2 \cdot 1}}{\cancel{(4 \cdot 3 \cdot 2 \cdot 1)}(1)} \\ &= 1\end{aligned}$$

The fact that $0! = 1$ makes this formula computable. It gives the correct value because a set of size 4 has exactly one subset of size 4, namely itself.

$$\mathbf{c.} \binom{n+1}{n} = \frac{(n+1)!}{n!((n+1)-n)!} = \frac{(n+1)!}{n!1!} = \frac{(n+1) \cdot \cancel{n!}}{\cancel{n!}} = n+1$$



Sequences in Computer Programming



Sequences in Computer Programming

An important data type in computer programming consists of finite sequences. In computer programming contexts, these are usually referred to as *one-dimensional arrays*.

For example, consider a program that analyzes the wages paid to a sample of 50 workers.

Such a program might compute the average wage and the difference between each individual wage and the average.



Sequences in Computer Programming

This would require that each wage be stored in memory for retrieval later in the calculation.

To avoid the use of entirely separate variable names for all of the 50 wages, each is written as a term of a one-dimensional array:

$$W[1], W[2], W[3], \dots, W[50].$$



Example 18 – *Dummy Variable in a Loop*

The index variable for a **for-next** loop is a dummy variable. For example, the following three algorithm segments all produce the same output:

1. **for** $i := 1$ **to** n
 print $a[i]$
next i

2. **for** $j := 0$ **to** $n - 1$
 print $a[j + 1]$
next j

3. **for** $k := 2$ **to** $n + 1$
 print $a[k - 1]$
next k



Sequences in Computer Programming

The recursive definitions for summation, product, and factorial lead naturally to computational algorithms.

For instance, here are two sets of pseudocode to find the sum of $a[1]$, $a[2]$, ..., $a[n]$.

The one on the left exactly mimics the recursive definition by initializing the sum to equal $a[1]$; the one on the right initializes the sum to equal 0.



Sequences in Computer Programming

In both cases the output is $\sum_{k=1}^n a[k]$.

```
s := a[1]
```

```
for k := 2 to n
```

```
    s := s + a[k]
```

```
next k
```

```
s := 0
```

```
for k := 1 to n
```

```
    s := s + a[k]
```

```
next k
```



Application: Algorithm to Convert from Base 10 to Base 2 Using Repeated Division by 2



Application: Algorithm to Convert from Base 10 to Base 2 Using Repeated Division by 2

A systematic algorithm to convert any nonnegative integer to binary notation uses repeated division by 2.

Suppose a is a nonnegative integer. Divide a by 2 using the quotient-remainder theorem to obtain a quotient $q[0]$ and a remainder $r[0]$. If the quotient is nonzero, divide by 2 again to obtain a quotient $q[1]$ and a remainder $r[1]$.

Continue this process until a quotient of 0 is obtained. At each stage, the remainder must be less than the divisor, which is 2. Thus each remainder is either 0 or 1.

Application: Algorithm to Convert from Base 10 to Base 2 Using Repeated Division by 2

The process is illustrated below for $a = 38$. (Read the divisions from the bottom up.)

					0	remainder = 1 = $r[5]$
				2	1	remainder = 0 = $r[4]$
			2	2		remainder = 0 = $r[3]$
		2	4			remainder = 1 = $r[2]$
	2	9				remainder = 1 = $r[1]$
	2	19				remainder = 0 = $r[0]$
2	38					

The results of all these divisions can be written as a sequence of equations:

$$38 = 19 \cdot 2 + 0,$$



Application: Algorithm to Convert from Base 10 to Base 2 Using Repeated Division by 2

$$19 = 9 \cdot 2 + 1,$$

$$9 = 4 \cdot 2 + 1,$$

$$4 = 2 \cdot 2 + 0,$$

$$2 = 1 \cdot 2 + 0,$$

$$1 = 0 \cdot 2 + 1.$$

By repeated substitution, then,

$$38 = 19 \cdot 2 + 0$$

$$= (9 \cdot 2 + 1) \cdot 2 + 0 = 9 \cdot 2^2 + 1 \cdot 2 + 0$$

$$= (4 \cdot 2 + 1) \cdot 2^2 + 1 \cdot 2 + 0 = 4 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2 + 0$$

$$= (2 \cdot 2 + 0) \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2 + 0$$



Application: Algorithm to Convert from Base 10 to Base 2 Using Repeated Division by 2

$$\begin{aligned} &= 2 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2 + 0 \\ &= (1 \cdot 2 + 0) \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2 + 0 \\ &= 1 \cdot 2^5 + 0 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2 + 0. \end{aligned}$$

Note that each coefficient of a power of 2 on the right-hand side is one of the remainders obtained in the repeated division of 38 by 2.

This is true for the left-most 1 as well, because $1 = 0 \cdot 2 + 1$. Thus

$$38_{10} = 100110_2 = (r[5]r[4]r[3]r[2]r[1]r[0])_2.$$



Application: Algorithm to Convert from Base 10 to Base 2 Using Repeated Division by 2

In general, if a nonnegative integer a is repeatedly divided by 2 until a quotient of zero is obtained and the remainders are found to be $r[0], r[1], \dots, r[k]$, then by the quotient-remainder theorem each $r[i]$ equals 0 or 1, and by repeated substitution from the theorem,

$$a = 2^k \cdot r[k] + 2^{k-1} \cdot r[k-1] + \dots + 2^2 \cdot r[2] + 2^1 \cdot r[1] + 2^0 \cdot r[0]. \quad 5.1.5$$

Thus the binary representation for a can be read from equation (5.1.5):

$$a_{10} = (r[k]r[k-1] \dots r[2]r[1]r[0])_2.$$



Example 19 – Converting from Decimal to Binary Notation Using Repeated Division by 2

Use repeated division by 2 to write the number 29_{10} in binary notation.

Solution:

				0	remainder = $r[4] = 1$
			2	1	remainder = $r[3] = 1$
		2	3		remainder = $r[2] = 1$
	2	7			remainder = $r[1] = 0$
2	14				remainder = $r[0] = 1$
2	29				

Hence $29_{10} = (r[4] r[3] r[2] r[1] r[0])_2 = 11101_2$.



Application: Algorithm to Convert from Base 10 to Base 2 Using Repeated Division by 2

The procedure we have described for converting from base 10 to base 2 is formalized in the following algorithm:

Algorithm 5.1.1 Decimal to Binary Conversion Using Repeated Division by 2

[In this Algorithm the input is a nonnegative integer n . The aim of the algorithm is to produce a sequence of binary digits $r[0]$, $r[1]$, $r[2]$, . . . , $r[k]$ so that the binary representation of n is

$$(r[k]r[k-1]\cdots r[2]r[1]r[0])_2.$$

That is,

$$n = 2^k \cdot r[k] + 2^{k-1} \cdot r[k-1] + \cdots + 2^2 \cdot r[2] + 2^1 \cdot r[1] + 2^0 \cdot r[0].]$$



Application: Algorithm to Convert from Base 10 to Base 2 Using Repeated Division by 2

Input: n [*a nonnegative integer*]

Algorithm Body:

$q := n, i := 0$

[Repeatedly perform the integer division of q by 2 until q becomes 0. Store successive remainders in a one-dimensional array $r[0], r[1], r[2], \dots, r[k]$.

Even if the initial value of q equals 0, the loop should execute one time (so that $r[0]$ is computed).

*Thus the guard condition for the **while** loop is $i = 0$ or $q \neq 0$.]*



Application: Algorithm to Convert from Base 10 to Base 2 Using Repeated Division by 2

while ($i = 0$ or $q \neq 0$)

$r[i] := q \bmod 2$

$q := q \operatorname{div} 2$

[$r[i]$ and q can be obtained by calling the division algorithm.]

$i := i + 1$

end while



Application: Algorithm to Convert from Base 10 to Base 2 Using Repeated Division by 2

[After execution of this step, the values of $r[0]$, $r[1]$, ..., $r[i-1]$ are all 0's and 1's, and $a = (r[i-1] r[i-2] \dots r[2] r[1] r[0])_2$.]

Output: $r[0]$, $r[1]$, $r[2]$, ..., $r[i-1]$ *[a sequence of integers]*