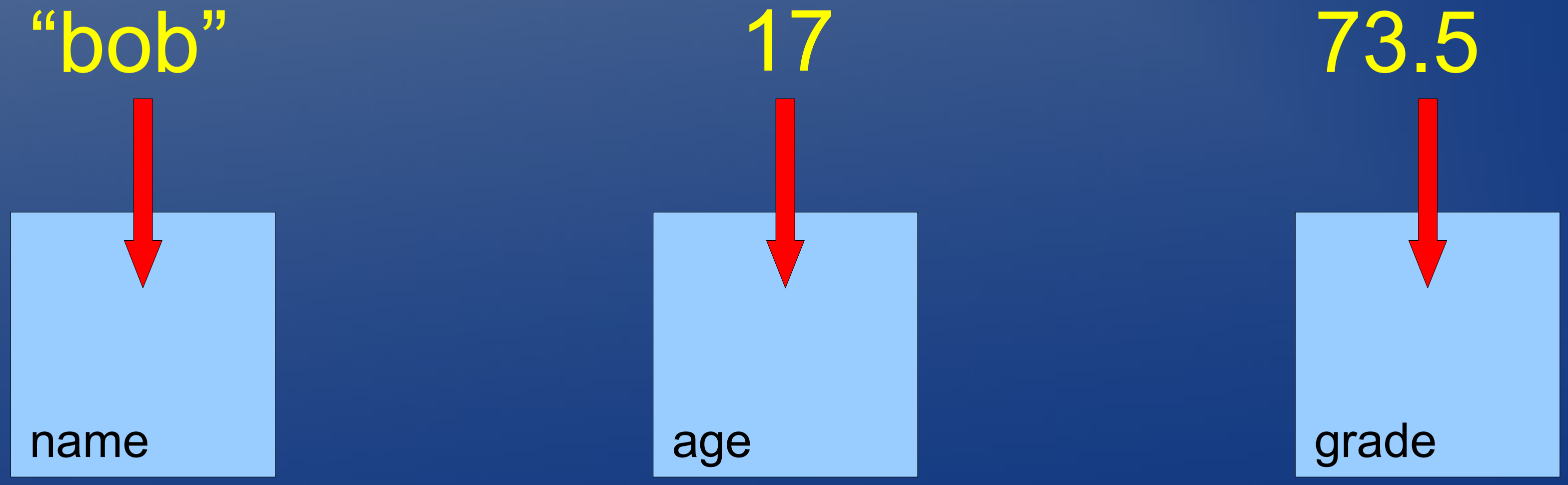
Introduction to Arrays

Why Variables?

Each time we declare a variable, we are reserving a space in the memory of the computer. As the program runs (or executes), this space will be used to store useful information.



Limits on Variables

As programs gets larger and more complex, the number of variables also increases.

It is possible to recycle variables: When the variable is no longer used for one purpose, we can use it again for another purpose (with the same data type).

Even with recycling, the number of variables can get prohibitively large.

How to Make a Program More Manageable?

One way to deal with complicated code is to break it into smaller pieces. This can be done with:

A. Subprograms
   1. Functions
   2. Procedures

B. “include” files

C. Arrays

Collections of Data

Another way to organize and simplify our programs is to make collections of identical data types that have the same description.

For example, suppose you were writing a grading program, but instead of a term (4 grades), it was the entire set of grades for high school.

How many variables would you need?

Introducing the Array

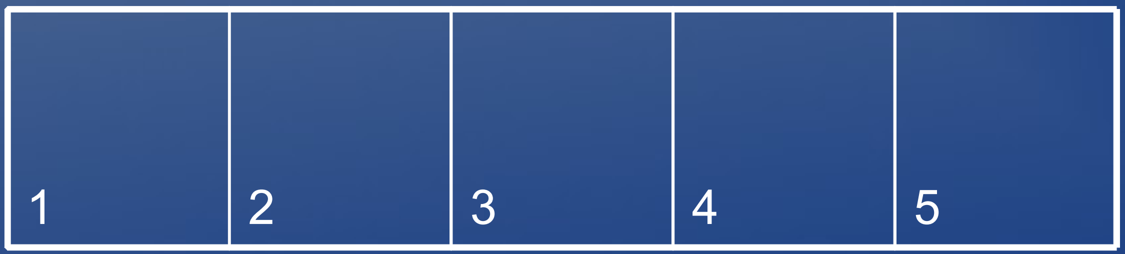
The array is a special data structure that allows us to make large collections of data that:

(a) are of the same data type
    (i.e., int, real, string, etc...)
(b) will be used for the same purpose
    (e.g., grades, names, ages, addresses, etc...)

Arrays – a Collection of Variables

The common way to draw or visualize the array is using a group of connected boxes. Each box in the array has a position (1st, 2nd, 3rd, etc...).

Each box is called an element of the array, and the position of each element is the index.



an array with 5 elements

Arrays in Turing

To declare an array in Turing:

```
var name : array low .. high of dataType
```

name – the name of the array

low – the lower index value (usually 1 or 0)
high – the upper index value

dataType – integer, string, real, etc...

Some Sample Array Declarations

```
var names : array 1 .. 10 of string

var savings : array 1 .. 100 of real

var year : array 1990 .. 2010 of int
```

Using the Array

To access the data in an array, you must give the name of the array variable and the index of the data item.

```
var firstname : array 1 .. 3 of string

% initialize the array
firstname(1) := “fred”
firstname(2) := “wilma”
firstname(3) := “pebbles”

% output the array
for i : 1 .. 3
   put firstname(i)
end for
```