module3

gar

Outline

Functions, Arrays and Strings

Arrays

- Collection of elements of same data type
- When declaring, square brackets are used to indicate that data is an array
  - E.g. If marks of 10 students are to be stored

    ```
    int marks[10];
    ```

  - To get the marks of first student

    ```
    printf("%d\n", a[0]);
    ```

  - To set the marks of the 10^{th} student as 65

    ```
    a[9] = 65;
    ```

  - We may also declare an array of roll numbers, to know who the n^{th} student is

    ```
    int rollno[10];
    ```

  - It's also possible to logically group the `rollno` and `marks` as a structure, which is a topic of module 4

Concept of an Array

- Each number 0, 1, 2 . . . 9 is called **subscript** or **index**
- Square brackets are used for indexing
- Indexing starts from 0 and goes up to n-1, where n is the maximum size reserved for an array
- Initializing an array

```
int a[5] = {-2,-1,0,1,2};
```

  - Can also be set individually after declaring

```
int a[5];
a[0] = -2;
a[1] = -1;
```

- If only one element is Initialized during declaration, all others will be set to zeroes

```
int a[5] = {-2,-1};
```

is same as

```
int a[5] = {-2,-1,0,0,0};
```

Programming Example using Array

- Read 5 floating point values to an array a and then compute the sum

```
main()
{
  float a[5], sum = 0;
  int i;
  for (i=0; i<5; i++)
    scanf("%f", &a[i]);
  for (i=0; i<5; i++)
    sum += a[i];
}
```

- Extend it to find
  1. Mean $= \bar{a} = \frac{\sum_{i=0}^{4} a_i}{5}$
  2. Variance $= \sigma^2 = \frac{\sum_{i=0}^{4} (a_i - \bar{a})^2}{5}$

Array of *char*

- Array of characters is also called a string
- Each element can store the ASCII code of the character
- Every string must end with a null character, which is \0
  - When printing the string, all the characters till \0 will be printed
- E.g.

```
char quote1[20] = "Hello";
char quote2[20] = {'H','e','l','l','o','\0'};
char quote3[20];
quote3[0] = 'H';
quote3[1] = 'e';
quote3[2] = 'l';
quote3[3] = 'l';
quote3[4] = 'o';
quote3[5] = '\0';
```

- quote1 and quote2 mean exactly the same. But in quote3, we need to specify the null character explicitly

Printing and Reading Strings

- Use the conversion specifier %s when reading and printing

```
printf("%s", quote1);
```

- & operator is not required in this case, since specifying the array name implies the address of the string.

```
scanf("%s", quote2);
```

Function

- Provides a convenient way to encapsulate a computation
- Can later be used without worrying about its implementation
  - `printf()`, `sqrt()` etc.
- E.g. Define a function to calculate power (x^y)

```
int power(int x, int y)
{
  int i, p = 1;
  for (i = 1; i <= y; i++)
    p = p*x;
  return p;
}
```

Syntax of function definition

```
return-type function-name(parmeter declarations, if any)
{
  declarations
  statements
}
```

- Function definitions may appear
  - in any order
  - in one source file or several