Lecture 1:
Arrays (Part 1)

CS102-Computer Programming 2

2nd Semester 2023-2024

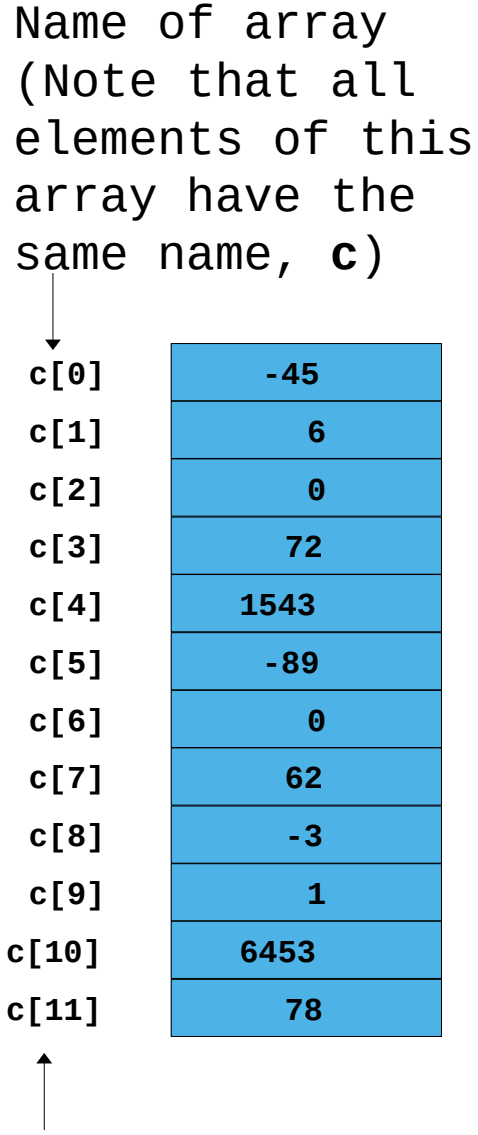
Introduction

- Arrays
  - Structures of related data items
  - Static entity – same size throughout program
  - Dynamic data structures

Arrays

- Array
  - Group of consecutive memory locations
  - Same name and type
- To refer to an element, specify
  - Array name
  - Position number
- Format:

  arrayname **[** *position number* **]**
  - First element at position **0**
  - **n** element array named **c**:
    - **c[0]**, **c[1]**…**c[n – 1]**



Arrays

- Array elements are like normal variables

```
c[ 0 ] =  3;
printf( "%d", c[ 0 ] );
```

  - Perform operations in subscript. If **`x`** equals **`3`**

```
c[ 5 - 2 ] == c[ 3 ] == c[ x ]
```

Declaring Arrays

- When declaring arrays, specify
  - Name
  - Type of array
  - Number of elements

```
arrayType arrayName[ numberOfElements ];
```

  - Examples:

```
int c[ 10 ];
float myArray[ 3284 ];
```

- Declaring multiple arrays of same type
  - Format similar to regular variables
  - Example:

```
int b[ 100 ], x[ 27 ];
```

Examples Using Arrays

- Initializers

```
int n[ 5 ] = { 1, 2, 3, 4, 5 };
```

  - If not enough initializers, rightmost elements become **0**

```
int n[ 5 ] = { 0 }
```

    - All elements 0
  - If too many a syntax error is produced
  - C arrays have no bounds checking
- If size omitted, initializers determine it

```
int n[ ] = { 1, 2, 3, 4, 5 };
```

  - 5 initializers, therefore 5 element array

Example: MonthlyRainfall

Problem: using *Rainfall Table*

- input month
- output mean rainfall for that month

month	mean rainfall (in mm)
0	30
1	40
2	45
3	95
4	130
5	220
6	210
7	185
8	135
9	80
10	40
11	45

Rainfall Table

Example (cont): MonthlyRainfall (v.1)

```
#include <stdio.h>

int main()
{
  int    month;
  int    table[12] = { 30, 40, 45, 95, 130, 220,
                       210, 185, 135, 80, 40, 45 };

  printf("Enter month: ");
  scanf("%d", &month);

  printf("Average rainfall: %d mm.\n", table[month-1]);

  return 0;
}
```

rainfall1.c

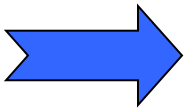
Example (cont): MonthlyRainfall (v.1)

```
#include <stdio.h>

int main()
{
  int    month;
  int    table[12] = { 30, 40, 45, 95, 130, 220,
                       210, 185, 135, 80, 40, 45 };

  printf("Enter month: ");
  scanf("%d", &month);

  printf("Average rainfall: %d mm.\n", table[month-1]);

  return 0;
}
```

rainfall1.c

Input / Output of Arrays

- Library functions printf() and scanf() do not know about arrays



So we have to do I/O ourselves

Example: IORainfall-1

```
#include <stdio.h>
#define  NMONTHS 12

/* Store and print rainfall */

int main()
{
  int data[NMONTHS];
  int month;

  for ( month=0; month < NMONTHS; month++ )
  {
     scanf("%d", &data[month] );
  }

  ...
```

rainio1.c

Example (cont): IORainfall-1

```
#include <stdio.h>
#define  NMONTHS 12

/* Store and print rainfall */

int main()
{
  int data[NMONTHS];
  int month;

  for ( month=0; month < NMONTHS; month++ )
  {
     scanf("%d", &data[month] );
  }

  ...
```

rainio1.c

Example (cont): IORainfall-2 (v.1)

```
#include <stdio.h>
#define  NMONTHS 12
 ...
  /* Print from January to December */
  for ( month=0; month < NMONTHS; month++ )
  {
     printf( "%d ", data[month] );
  }
  printf("\n");

  /* Print from December to January */
  for ( month = NMONTHS - 1; month >= 0; month-- )
  {
    printf( "%d ", data[month] );
  }
  printf("\n");
  return 0;
}
```

rainio1.c

Example (cont): IORainfall-2 (v.1)

```
#include <stdio.h>
#define  NMONTHS 12
 ...
  /* Print from January to December */
  for ( month=0; month < NMONTHS; month++ )
  {
     printf( "%d ", data[month] );
  }
  printf("\n");

  /* Print from December to January */
  for ( month = NMONTHS - 1; month >= 0; month-- )
  {
    printf( "%d ", data[month] );
  }
  printf("\n");
  return 0;
}
```

rainio1.c

Example (cont): IORainfall-2 (v.2)

```
#include <stdio.h>
#define  NMONTHS 12
...
  /* Print from January to December */
  for ( month=0; month < NMONTHS; month++ )
  {
     printf( "%5d ” , data[month] );
  }
  printf("\n");

  /* Print from December to January */
  for ( month = NMONTHS - 1; month >= 0; month-- )
  {
    printf( "%5d ” , data[month] );
  }
  printf("\n");

  return 0;
}
```

rainio2.c

Handling Indices

- Arrays have a fixed size
- There is no built-in way of checking if the supplied index is within range
- We must check for valid indices ourselves

Example (cont): MonthlyRainfall (v.2)

```c
#include <stdio.h>
#define MAXLEN 1024
int main()
{
  int    month;
  char   line[MAXLEN];
  char   dummy[MAXLEN];
  int table[12] = { 30, 40, 45, 95, 130, 220, 210, 185, 135, 80, 40, 45 };

  while(1)
  {
    printf("Enter month or ctrl-c to end: ");
    fgets(line, MAXLEN, stdin);
    if (sscanf(line, "%d%s", &month, dummy) != 1)   /* valid input? */
    {
      printf("Invalid input. Try again.\n");
    }
    else if (1 <= month && month <= 12)             /* input in range? */
    {
      printf("Average rainfall for month %d is %d mm.\n",month,table[month-1]);
    }
    else
    {
      printf("Month should be between 1 and 12. Try again.\n");
    }
  }
  return 0;
}
```

rainfall2.c

Example (cont): MonthlyRainfall-1 (v.3)

```
#include <stdio.h>
#define MAXLEN 1024
int rainfall(int month);
/* Main program to test rainfall() function */
int main()
{
  int    month;
  char   line[MAXLEN];
  char   dummy[MAXLEN];
  while(1)
  {
    printf("Enter month or ctrl-c to end: ");
    fgets(line, MAXLEN, stdin);
    if (sscanf(line, "%d%s", &month, dummy) != 1)
    {
      printf("Invalid input. Try again.\n");
    }
    else if (1 <= month && month <= 12)
    {
      printf("Average rainfall for month %d is %d mm.\n",month,rainfall(month-
1));
    }
    else
    {
      printf("Month should be between 1 and 12. Try again.\n");
    }
  }
  return 0;
}
```

rainfall3.c

Example (cont): MonthlyRainfall-2 (v.3)

```
/*************************************************************\
 * NAME:
 *    int rainfall(int month)
 * DESCRIPTION:
 *    Returns the mean monthly rainfall (in millimeters)
 *    in a given month
 * PRE:
 *    The integer `month' must be between 0 and 11, where
 *    0 = January, 1 = February, etc.  Otherwise, the behaviour
 *    is undefined
 *    The local array `table' should be initialized to contain
 *    the average rainfall in a given month
 * POST:
 *    It returns an integer value corresponding to the mean
 *    rainfall (in millimeters) for the given `month'
\*************************************************************/
int rainfall ( int month )
{
  int table[12] = { 30, 40, 45, 95, 130, 220,
                    210, 185, 135, 80, 40, 45 };
  return (table[month]);
}
```

rainfall3.c

Example (cont): MonthlyRainfall-2 (v.3)

```
/*************************************************************\
 * NAME:
 *    int rainfall(int month)
 * DESCRIPTION:
 *    Returns the mean monthly rainfall (in millimeters)
 *    in a given month
 * PRE:
 *    The integer `month' must be between 0 and 11, where
 *    0 = January, 1 = February, etc.  Otherwise, the behaviour
 *    is undefined
 *    The local array `table' should be initialized to contain
 *    the average rainfall in a given month
 * POST:
 *    It returns an integer value corresponding to the mean
 *    rainfall (in millimeters) for the given `month'
\*************************************************************/
int rainfall ( int month )
{
  int table[12] = { 30, 40, 45, 95, 130, 220,
                    210, 185, 135, 80, 40, 45 };
  return (table[month]);
}
```

rainfall3.c

Passing Arrays to Functions

- The array is passed
  - as an array of unspecified size (`int array[]`) OR
  - as a pointer (`int *array`)
- Changes to the array within the function affect the "original" array

Example (cont): IORainfall-1 (v.3)

```
#include <stdio.h>
#define  NMONTHS 12

void loadRain ( int arrayPtr[] )
{
  int month;

  for (month=0; month < NMONTHS; month++)
  {
     scanf("%d", &arrayPtr[month]);
  }
}
```

rainio3.c

Example (cont): IORainfall-2 (v.3)

```
void printRain ( const int arrayPtr[] )
{
  int month;

  for (month=0; month < NMONTHS; month++)
  {
     printf("%5d", arrayPtr[month]);
  }

  printf("\n");
}
```

rainio3.c

Example (cont): IORainfall-3 (v.3)

```c
#include <stdio.h>
#define  NMONTHS 12

void loadRain ( int arrayPtr[] );
void printRain ( const int arrayPtr[] );

/* Store and print rainfall */
int main()
{
  int data[NMONTHS];

  loadRain(data);
  printRain(data);

  return 0;
}
```

rainio3.c

Example: IORainfall -- v.3 (cont)

```
#include <stdio.h>
#define  NMONTHS 12
void loadRain ( int arrayPtr[] );
void printRain ( const int arrayPtr[] );
/* Store and print rainfall */
int main()
{
  int data[NMONTHS];
  loadRain(data);
  printRain(data);
  return 0;
}
/* Read in rainfall for each month*/
void loadRain ( int arrayPtr[] )
{
  int month;
  for (month=0; month < NMONTHS; month++)
  {  scanf("%d", &arrayPtr[month]);  }
}
/* Print rainfall for each month*/
void printRain ( const int arrayPtr[] )
{
  int month;
  for (month=0; month < NMONTHS; month++)
  {  printf("%5d", arrayPtr[month]);  }
  printf("\n");
}
```

rainio3.c

End of Lecture 1