IS12 - Introduction to Programming  
Lecture 22: 2-dimensional Arrays

Peter Brusilovsky  
http://www2.sis.pitt.edu/~peterb/0012-072/

Outline

- 2-dimensional Arrays
- Example: Naval Battle
- Example: Day of the Month
- Arrays of pointers
- Example: Text Reversing
Two-Dimensional Arrays

- Two-dimensional array is simply a table where each cell has two addresses - row and column.
- The way to address an element of a 2-dimensional array \( a \) is \( a[\text{row}][\text{column}] \)
- Rows and Columns are numbered from 0

```
int a[5][5];
```

Declaration and Access

- Declaration with no initialization
  ```
  int battlefield[10][10];
  ```
- Declaration with initialization
  ```
  int battlefield[5][5] = {
    {1, 0, 1, 0, 0},
    {0, 0, 0, 0, 1},
    {0, 0, 1, 0, 0},
    {0, 0, 0, 0, 0},
    {1, 0, 0, 0, 1},
  };
  ```
- Access
  ```
  battlefield[4][4] = 1;
  ```
Example: Battleship (1)

```c
#include <stdio.h>
#define FSIZE 5
#define SHIP 1
#define EMPTY 0
#define DESTROYED 2

static int battlefield[FSIZE][FSIZE] = {
    {1, 0, 1, 0, 0}, /* row A */
    {0, 0, 0, 0, 1}, /* row B */
    {0, 0, 1, 0, 0}, /* row C */
    {0, 0, 0, 0, 0}, /* row D */
    {1, 0, 0, 0, 1}, /* row E */
};
```

```c
void main()
{
    int col;
    char ch_row;

    printf("Enter your move like 5 A\n");
    printf("Enter 0 X to exit\n");
    /* reads and process hits */
    do{
        printf("You move: ");
        scanf("%d %c\n", &col, &ch_row);
    } while(react(ch_row, col) > 0);
}
```

Example: Battleship (2)
Example: Battleship (3)

```c
int react(char r, int col) {
    int row;
    if (col == 0) {
        printf("Good Bye!\n"); return 0;
    }
    row = r - 'A';
    if (row < 0 || row > FSIZE - 1)
        printf("Row %c is not possible!\n", r);
    else if (col < 1 || col > FSIZE)
        printf("Column %d is not possible!\n", col);
    else if (battlefield[row][col-1] == SHIP) {
        printf("Hit!\n");
        battlefield[row][col-1] = DESTROYED;
    } else if (battlefield[row][col-1] == DESTROYED)
        printf("This ship has been already destroyed!\n");
    else 
        printf("Misss...\n");
    return 1;
}
```

Example: Day of the Year (1)

101 of 2000?

Jan  Feb  Mar  Dec
31  29  31  31  31

101 - 31 = 70
70 - 29 = 41
41 - 31 = 10

101 = April 10, 2000
Example: Day of the Year (2)

```c
static char daytab[2][13] = {
    {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31},
    {0, 31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31},
};

void month_day(int year, int yearday, int *pmonth, int *pday)
{
    int i, leap /* 1 for leap year, 0 for non-leap */;
    leap = ((year%4 == 0) && year%100 != 0) || year%400 == 0;
    for(i = 1; yearday > daytab[leap][i]; i++)
        yearday -= daytab[leap][i];
    *pmonth = i;
    *pday = yearday;
}
```

Example: Day of the Year (3)

```c
void main()
{
    int year, yday; /* input data */
    int month, mday; /* output data */
    do{
        printf("Enter Year and Day (yyyy dd): ");
        scanf("%d %d", &year, &yday);
        if(yday > 0) {
            month_day(year, yday, &month, &mday);
            printf("Day %d of %d is simply %d/%d/%d\n", yday, year, month, mday, year);
        } else
            printf("Good Bye!\n");
    } while(yday > 0);
}
```
Two-Dimensional Arrays Again

- Two-dimensional array is really an array of one-dimensional arrays
- We can address a whole one-dimensional array as a[row]
- A continuous segment of memory is allocated for a 2-dimensional array

Arrays of Pointers

char *strings[4]; /* array of four pointers */
Memory Allocation: `malloc`

```c
p1 = malloc(6);
```

```
p2 = malloc(8);
```

Flowchart showing the memory allocation for `malloc` with two pointers `p1` and `p2`.
Example: Reverse Text (1)

```c
static char *lineptr[MAXLINES];
main() {
    int nlines; /* counter of input lines read */
    nlines = readlines();
    printbackwards(nlines);
}

void printbackwards(int n)
{
    while(n > 0) {
        printf("%s", lineptr[n-1]);
        --n;
    }
}
```

Example: Reverse Text (2)

```c
#define MAXLEN 1000
int readlines()
{
    int len, nlines;
    char *p, line[MAXLEN];

    nlines = 0;
    while((len = getline(line, MAXLEN)) > 0) {
        p = (char *) malloc(len+1);
        copy(p, line);
        lineptr[nlines] = p;
        ++ nlines;
    }
    return nlines;
}
Initialization of Pointer Arrays

```c
static char *strings[4] = {
    "abc", "def", "gh"
};

static char *picture[3] = {
    "   ",
    "[ ]",
    "[X]"
};
```

Example: More Battleship

```c
static char *picture[3] = {"   ", "[ ]", "[X]"};
void printfield(char *header)
{
    int i, j;
    /* print header */
    printf("%s

   ", header);
    for(j = 0; j < FSIZE; ++j)
        printf(" %d  ", j+1);
    printf("\n");
    /* print field */
    for(i = 0; i < FSIZE; ++i) {
        printf("%c  ", i + 'A');
        for(j = 0; j < FSIZE; ++j)
            printf("%s ", picture[ battlefield[i][j] ]);  
        printf("\n");
    }
    printf("\n");
}
```
Before Next Lecture:

- Do reading assignment
- **Perry**: Chapter 25. Chapter 26
- **Ted Jensen**: Pointers and Arrays in C (http://home.netcom.com/~tjensen/ptr/cpoint.htm): Chapters 6-9
- Run Classroom Examples
- Use Knowledge Sea to find and read other relevant material on 2-d arrays and arrays pointers
- Exercise: enhance Day of the Year program with names of the months