Parallel Arrays

Parallel Arrays

- Data at the same index represent a concept
- Operations on arrays are carried out concurrently

names	addresses	marks
Joseph	12 Le Loi, Q1, TPHCM	7
Dinh Tan Vu	12/66 duong so 3, Go Vap, TPHCM	8
Miranda	123 Calmette, District 1, HCM City	5
Celine Dion	124 street 8, district 7, HCM City	9

Problem 1

- Data about an employee: Code(char 8), name (char 20), salary(double), allowance(double)
- Develop a C-program that allows user:
  - Adding a new employee
  - Find data about employees using a name inputted.
  - Remove an employee based on a code inputted
  - Print the list in descending order based on salary + allowance.

Problem 1… Analysis

- Data:
  - Constant: MAXN = 50
  - 4 arrays for the employee list: char codes[MAXN][9], names[MAXN][21], double salaries[MAXN], double allowances[MAXN].
  - int n=0; /* current number of employees */
  - char code[9]; /* inputted code */
  - char name[21]; /* name inputted */
  - int choice; /* user choice */

Problem 1… Analysis

- Operations:

```
/* Getting a user choice */
int menu()
/* Add a new employee, inputted data are local variables */
void add (char codes[][9],char names[][21], double salaries[], double allowances[], int*pn)
/* Print out data about employees bases on a known name */
void printBasedName( char name[], char codes[][9],char names[][21], double salaries[],
        double allowances[], int n)
/* Find the position of a known code */
int findCode (char code[], char codes[][9], int n)
/* Remove the employee at the position pos */
void removePos (int pos, char codes[][9],char names[][21], double salaries[], double
        allowances[], int *pn)
/* Sort the list based on salary+allowance*/
void sort(char codes[][9],char names[][21], double salaries[], double allowances[], int n)
/* Print all the list to the monitor */
 void print(char codes[][9],char names[][21], double salaries[], double allowances[], int n)
```

Problem 1… Analysis

```
/* Sort the list based on salary + allowance*/
void sort(char codes[][9], char names[][21], double salaries[],
                    double allowances[], int n)
  {   for (i=0; i<n-1; i++)
        for (j=n-1; j>i; j-- )
          if ( salaries[j] + allowances[j] <   salaries[j-1] + allowances[j-1] )
               {    swap codes[j], codes[j-1];
                    swap names[j], names[j-1];
                    swap salaries[j], salaries[j-1];
                    swap allowances[j], allowances[j-1];
                }
          }
      }
```

Problem 2

- Data about an item: Code(char 8), name (char 20), price(double), category (char 12)
- Develop a C-program that allows user:
  - Adding a new item
  - Print out items which belong to a known category.
  - Remove an item based on a code inputted
  - Print the list in ascending order based on categories then names

Problem 2… Analysis

- Data:
  - Constant: MAXN = 50
  - 4 arrays for the item list: char codes[MAXN][9], names[MAXN][21], int prices[MAXN], char categories[MAXN][13].
  - int n=0; /* current number of items */
  - char category[13]; /* inputted category */
  - char code[9]; /* name inputted */
  - int choice; /* user choice */
- Operations:

Problem 2… Analysis

- Operations:

/* Getting a user choice */

int menu()

/* Add a new item, inputted data are local variables */

void add (char codes[][9], char names[][21], int prices[], char categories[][13], int*pn)

/* Print out items of a known category */

void printACategory(char cat[], char codes[][9], char names[][21], int prices[], char categories[][13], int n)

/* Find the position of a known code */

int findCode (char code[], char codes[][9], int n)

/* Remove the item at the position pos */

void removePos (int pos, char codes[][9], char names[][21], int prices[], char categories[][13], int* pn)

/* Sort the list based on categories then names*/

void sort(char codes[][9], char names[][21], int prices[], char categories[][13], int n)

/* Print all the list to the monitor */

void print(char codes[][9], char names[][21], int prices[], char categories[][13], int n)

Problem 2… Analysis

```
/* Sort the list based on categories then names*/
void sort(char codes[][9], char names[][21], int prices[],
                    char categories[][13], int n)
  {   for (i=0; i<n-1; i++)
          for (j=n-1; j>i; j-- )
           {   int dCat = strcmp( categories[j], categories[j-1]);  /* Category difference */
               int dName = strcmp( names[j], names[j-1]); /* name difference */
               if ( dCat<0 || (dCat==0 && dName <0))
                   {    swap codes[j], codes[j-1];
                        swap names[j], names[j-1];
                        swap prices[j], prices[j-1];
                        swap categories[j], categories[j-1];
                    }
             }
      }
```

Problem 3

- Data about a clock: make(char 20), color (char 20), price(int), guarantee (int- bảo hành)
- Develop a C-program that allows user:
  - Adding a new clock
  - Printing out clocks which belong to a known make.
  - Printing out clocks whose prices are between p1 and p2 (integers)
  - Printing the list in descending order based on prices.

Problem 4

- Data about a soft drink: name (char 20), make(char 20), volume (int), price(int), duration (int- number of days when this product can be drunk)
- Develop a C-program that allows user:
  - Adding a new soft drink
  - Printing out items which belong to a known make.
  - Printing out items whose volumes are between v1 and v2 (integers)
  - Printing the list in ascending order based on volumes then prices.