Fpt University

Slot 13
Programming With Menu

A review for C-Functions

Pointers are parameters of functions

Using some C++ characteristics

Why is Menu?

- Generally, a program performs some operations and at a time only one task is carried out. ➔ A menu is usually used.

➔ How are menus implemented in C program?

Idea

- ***Common Algorithm in the entry point***:

```
int userChoice;
do
{  userChoice= getUserChoice();
   switch (userChoice)
   {  case 1: function1(); break;
      case 2: function2(); break;
      case 3: function3(); break;
   }
}
while (userChoice >0 && userChoice<maxChoice);
```

Problem

- Write a C program using the following menu:

  1- Operation 1

  2- Operation 2

  Others- Quit
- If user chooses 1, user will input 2 integers, the program will print out sum of integers between them including them.
- If user chooses 2, user will input 2 characters, the program will print out the ASCII table between two inputted characters in ascending order.
- If user chooses other options, the program will terminate.

Implementation: menuDemo1.c

```c
/* MenuDemo1.c */
#include <stdio.h>
/* Function for getting a choice from user - Menu */
int getUserChoice()
{ int choice; /* choice from user */
  /* print out the meu */
  printf("\n1-Operation 1");
  printf("\n2-Operation 2");
  printf("\nOthers-Quit");
  /* Accept user choice */
  printf("\nChoose:");
  /* %*c: Remove the ENTER key but no character variable
  is needed*/
  scanf("%d%*c", &choice);
  return choice;
}
```

Implementation: menuDemo1.c

```c
/* Function for operation 1
   user will input 2 integers, the program will print out
   sum of integers between them including them
*/
int sumBetween(int a, int b)
{   int t;
    if (a>b) /* a must be less than
    {  t= a; a=b; b=t;
    }
    int S= 0;
    for (t=a; t<=b; t++) S+=t;
    return S;
}
void function1()
{   int n1, n2; /* 2 integers */
    printf("Enter 2 integers:");
    scanf("%d%d%*c", &n1, &n2);
    printf("Sum=%d\n",sumBetween(n1, n2));
}
```

```
1-Operation 1
2-Operation 2
Others-Quit
Choose:1
Enter 2 integers:9 5
Sum=35
```

Implementation: menuDemo1.c

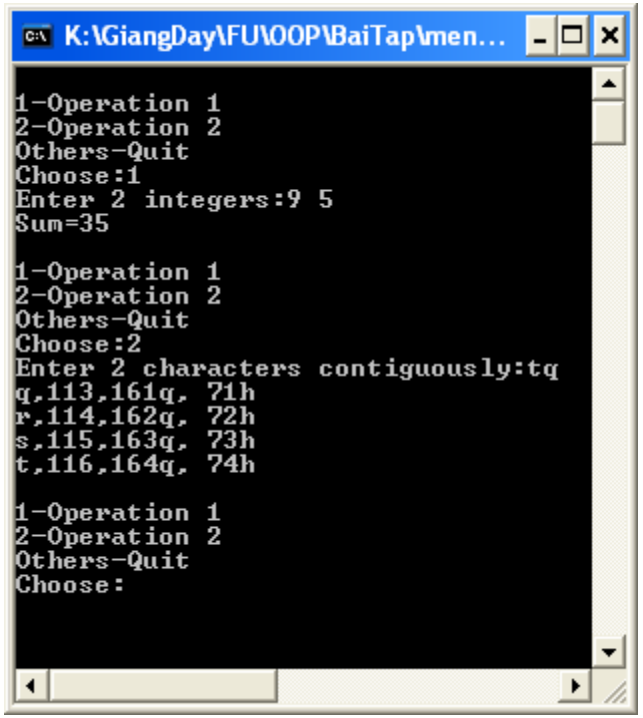
```c
/*  Operation 2:user will input 2 characters, the program
    will print out the ASCII table between two inputted
    characters in ascending order.
*/
/* Print ASCII table betwwen 2 characters,ascending order
void printAscii(char c1, char c2)
{   char c;
    if (c1>c2) /* c1 must be less than c2 */
    {   c=c1; c1=c2; c2=c;
    }
    for (c=c1; c<=c2; c++)
      printf("%c,%3d,%3oq,%3Xh\n", c,c,c,c);
}
void function2()
{   char c1, c2; /* inputted characters */
    printf("Enter 2 characters contiguously:");
    scanf("%c%c", &c1, &c2);
    printAscii(c1, c2);
}
```

```
1-Operation 1
2-Operation 2
Others-Quit
Choose:2
Enter 2 characters contiguously:tq
q,113,161q, 71h
r,114,162q, 72h
s,115,163q, 73h
t,116,164q, 74h
```

Implementation: menuDemo1.c

```c

int main()
{    int userChoice;
     do
     {   userChoice= getUserChoice();
         switch(userChoice)
         {   case 1: function1(); break;
             case 2: function2(); break;
             default: printf("Bye!\n");
         }
     }
     while (userChoice>0 && userChoice<3);
     fflush(stdin);
     getchar();
     return 0;
}
```



Functions with pointers as parameters

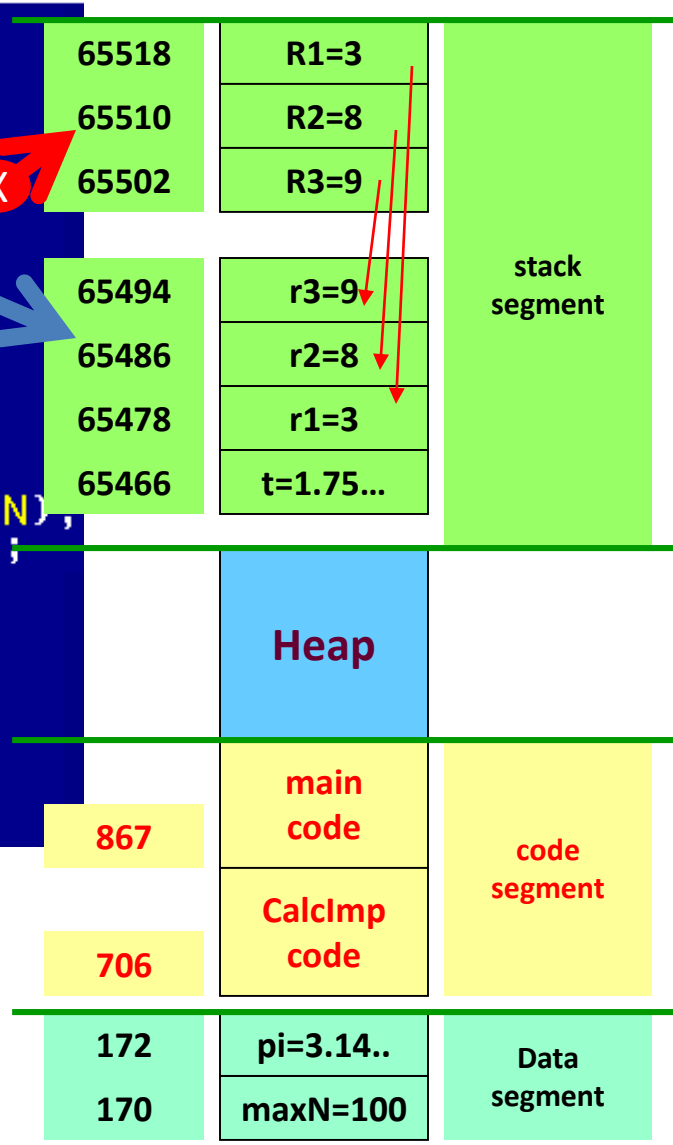
- C uses by-value parameters only → A function can not modify values of arguments.
- To modify values of arguments, pointers as parameters of a function are used.

Functions with pointers as parameters

Review: Pass by-value Parameters

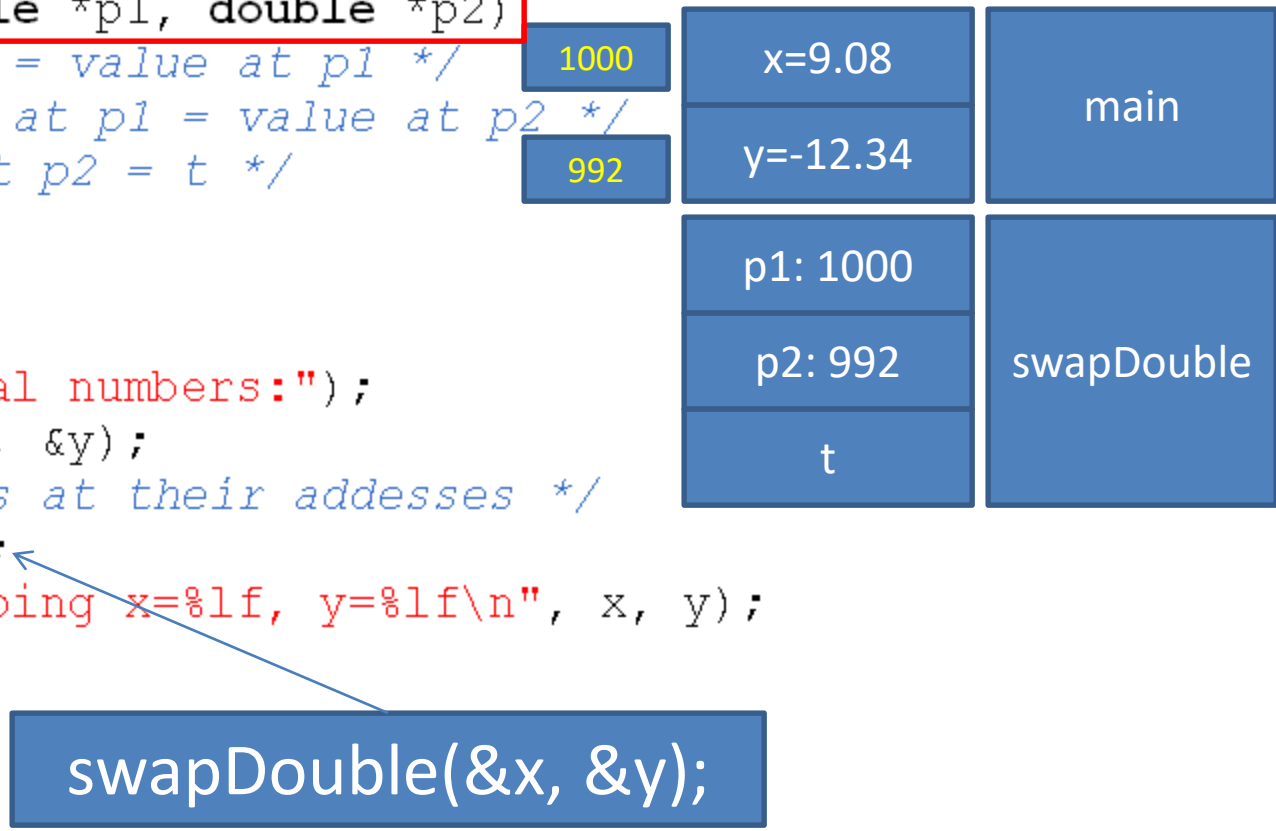
```
#include<stdio.h>
#include <conio.h>
int maxN=100;
double pi=3.141592;
double CalcImp(double r1, double r2, double r3)
{   double t=1/(1/r1 + 1/r2 + 1/r3);
    printf("r1    Addr:%u,value:%lf\n", &r1,r1);
    printf("r2    Addr:%u,value:%lf\n", &r2,r2);
    printf("r3    Addr:%u,value:%lf\n", &r3,r3);
    printf("t     Addr:%u,value:%lf\n", &t  ,t);
   return t;
}
int  main()
{ double R1=3, R2=8, R3=9;
   printf("maxN Addr:%u,  value:%d \n", &maxN,maxN);
   printf("pi   Addr:%u,  value:%lf\n", &pi   ,pi);
   printf("R1   Addr:%u,value:%lf\n", &R1   ,R1);
   printf("R2   Addr:%u,value:%lf\n", &R2   ,R2);
   printf("R3   Addr:%u,value:%lf\n", &R3   ,R3);
   printf("main    addr:%u\n", &main);
   printf("CalcImp addr:%u\n", &CalcImp);
   printf("Impedance: %lf",CalcImp(R1,R2,R3));
   getch();
}
```

```
maxN Addr:170,  value:100
pi   Addr:172,  value:3.141592
R1   Addr:65518,value:3.000000
R2   Addr:65510,value:8.000000
R3   Addr:65502,value:9.000000
main    addr:867
CalcImp addr:706
r1   Addr:65478,value:3.000000
r2   Addr:65486,value:8.000000
r3   Addr:65494,value:9.000000
t    Addr:65466,value:1.756098
Impedance: 1.756098
```



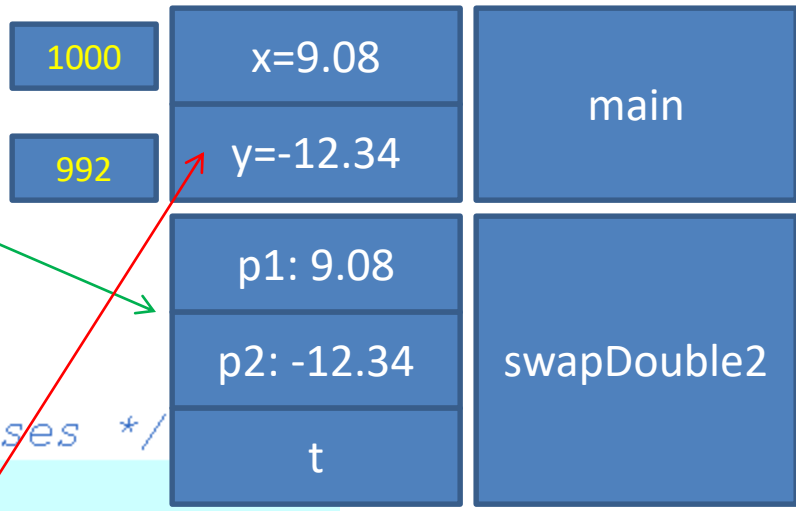
Pointers as parameters: Demo

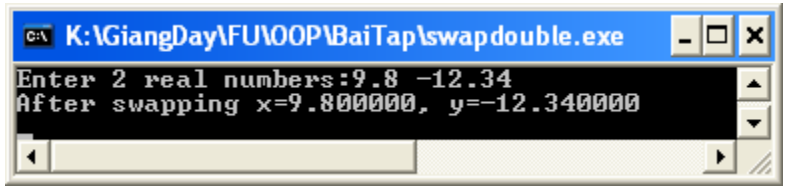
```c
/* Accept 2 numbers, swap them, then print out them */
#include <stdio.h>
/* SWAPPING 2 DOUBLE NUMBERS AT ADDRESSES p1, p2 */
void swapDouble (double *p1, double *p2)
{   double t=*p1; /* t = value at p1 */
    *p1= *p2; /* value at p1 = value at p2 */
    *p2= t; /* value at p2 = t */
}
int main()
{   double x, y;
    printf("Enter 2 real numbers:");
    scanf("%lf%lf", &x, &y);
    /* swaping 2 values at their addesses */
    swapDouble2 (x, y);
    printf("After swapping x=%lf, y=%lf\n", x, y);
    fflush(stdin);
    getchar();
    return 0;
}
```



Pointers as parameters: Demo

```
9  void swapDouble2 (double p1, double p2)
10 {   double t=p1;
11     p1= p2;
12     p2= t;
13 }
14
15 int main()
16 {   double x, y;
17     printf("Enter 2 real numbers:");
18     scanf("%lf%lf", &x, &y);
19     /* swaping 2 values at their addesses */
20     swapDouble2 (x, y);
21     printf("After swapping x=%lf, y=%lf\n", x, y);
22     fflush(stdin);
23     getchar();
24     return 0;
25 }
```





Introduction to C++

- C++ is an Object-Oriented Language
- It is developed from the C language and the language C is contained in C++ language
- The programming tool Dev-C++ supports both C and C++ source codes
- File extension of a C++ source code is **.cpp**
- We can use some C++ characteristics to develop programs more easily, such as:
  - **References** in C++ can be used as a replacement of pointers in function parameters
  - The **new** and **delete** operators to allocate/de-allocate dynamic data instead of C functions *malloc*, *calloc*, *free*
  - Utilities about variable declarations, comments

C++: References

- A way to give another name of a datum

Reference_Demo.cpp

```
/* References_de,o.cpp */
#include <stdio.h>
int main()
{   // in C, you must declare variables at the beginning of code
    int n1=10;
    printf("Variable n1, address: %u, vale:%d\n", &n1, n1);
    // in C++, you can declare variables freely
    int &n2= n1;
    printf("Variable n2, address: %u, vale:%d\n", &n2, n2);
    getchar();
}
```

// Comment to the line end

K:\GiangDay\FU\PFC\BTC-2015\Reference_Demo.exe

```
Variable n1, address: 2293620, vale:10
Variable n2, address: 2293620, vale:10
```

Both n1 and n2 are stored in only one memory block
→ n2 is the another name of n1

Function Parameters

You want	Use
Code of function can not modify arguments	Passing by value (characteristic of C)
Code of function can modify arguments	Pointers, **addresses** of arguments **cannot be modified** but **values in it can be modified by the operator ->**
	References of C++, names of arguments are passed to function

Passing Arguments

```
1  /* References_Demo2.cpp */
2  #include <stdio.h>
3  // Passing by values can not modify arguments
4  void swap1(int x, int y)
5  {   int t=x; x= y; y=t;
6  }
7  // Passing by values using pointers can modify arguments
8  void swap2(int* px, int* py)
9  {   int t=*px; *px= *py; *py=t;
10 }
11 // Passing by references can modify arguments
12 void swap3(int &x, int &y)
13 {   int t=x; x= y; y=t;
14 }
15 int main()
16 {   int a=3, b=4;
17     swap1(a,b);// passing by values
18     printf("a=%d, b=%d\n", a, b);
19     swap2(&a,&b);// passing by pointers(addresses)
20     printf("a=%d, b=%d\n", a, b);
21     int m= 7, n= 10;
22     swap3(m,n); // passing by references(names)
23     printf("m=%d, n=%d\n", m, n);
24     getchar();
25 }
```

```
a=3, b=4
a=4, b=3
m=10, n=7
```

Passing References to Function

```
/* References_Demo3.cpp */
// Input then ouput a dynamic array of integers
#include <stdio.h>
// Input an dynamic array of integers
void input(int* &ar, int& n)
{   printf("Number of elements:");
    scanf("%d", &n);
    ar= new int[n]; // dynamic allocation
    for (int i=0; i<n; i++)
    {   printf("Element %d :", i);
        scanf("%d", &ar[i]);
    }
}
// Output an array of integers
void output(int* ar, int n)
{   for (int i=0; i<n; i++) printf("%5d", ar[i]);
}
```

You can declare a local variable in the statement for

```
int main()
{   int* a=NULL, n;
    printf("Enter an array of integers:\n");
    input(a, n);
    printf("Values inputted:\n");
    output(a, n);
    delete a; // de-allocation
    getchar();
    getchar();
}
```

```
Enter an array of integers:
Number of elements:5
Element 0 :3
Element 1 :9
Element 2 :0
Element 3 :1
Element 4 :8
Values inputted:
    3    9    0    1    8
```