Module 3 – Pointer Types in C

Tutorial Questions

Objectives
To develop your understanding of pointer types and basic pointer arithmetic.

Activities

1. Explain the meaning of each of the following declarations:
    a. int *px;
    b. float a, b;
       float *pa, *pb;
    c. float a = -0.167;
       float *pa = &a;
    d. char c1, c2, c3;
       char *pc1, *pc2, *pc3 = &c1;
    e. double funct(double *a, double *b, int *c);
    f. double *funct(double *a, double *b, int *c);
    g. double *a[12];
    h. char *a[12];

2. Write an appropriate declaration for each of the following situations:
    a. Declare two pointers that point to the integer variables i and j.
    b. Declare a pointer to a floating-point quantity and a pointer to a double precision quantity.
    c. Declare a function that accepts two integer arguments and returns a pointer to a long integer.
    d. Declare a function that accepts two arguments and returns a long integer. Each argument will be a pointer to an integer quantity.
    e. Declare a one dimensional array of 10 pointers to floating-point quantities.

3. A C program contains the following declaration:

```
int x[8] = {10, 20, 30, 40, 50, 60, 70, 80};
```

    a. What is the meaning of x?
    b. What is the meaning of (x + 2)?
    c. What is the value of *x?
    d. What is the value of (*x + 2)?
    e. What is the value of *(x + 2)?

4. The skeletal structure of a C program is shown below:

```
void funct(int *p)
int main(void)
{
   int a[5] = {10, 20, 30, 40, 50};
   ..........
   funct(a+3, 2);
   ..........
}
```

```
void funct(int *p, int num)
{
   int i, sum = 0;

   for (i = 1; i < num; ++i)
         sum += *(p + i);

   printf("sum=%d\n", sum);
   return;
}
```

a. What kind of argument is passed to funct?
b. What kind of information is returned from funct?
c. What information is actually passed to funct?
d. What is the purpose of the for loop that appears within funct?
e. What value is displayed by the printf statement within funct?

5. Assume the following definitions:

```
typedef enum {clubs, diamonds, hearts, spades} Suit;
typedef struct card {
                              int pips;
                              Suit suit;
} Card;
```

Write a function `assignCard(Card *c, int p, Suit s)` which assigns pips value `p` and Suit value `s` to a `Card c`.
Assuming a local variable in main: `Card myCard;` give an example of main calling `assignCard` to assign values to `myCard`.