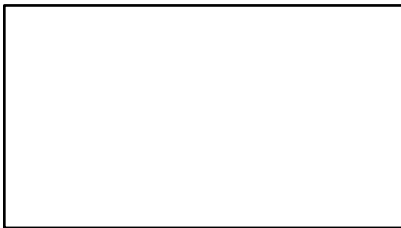

Tutorial - 8

Objectives: To practice with

- Pointers

1. Assuming the following declaration of the variables `cType` and `voltage`.
Declare the corresponding number of pointer type variables and assign them with the memory addresses of `cType` and `voltage`.

```
Char cType;  
float voltage;
```



2. Can one pointer be used to access several variables (one at a time) of the same type? Provide an example.



3. Provide an example of changing a variable value using a pointer and the indirection operator



4. What are the values of `ptrI` and `ptrC`?

```
int *ptrI = 1000;  
char *ptrC = 1000;
```

```
ptrI += 2;  
ptrC += 2;
```

Explain the results.

5. Given the declarations:

```
int m = 25, n = 77;
char c = '*';
int *itemp;
```

describe the errors in each of the following statements.

```
m = &n;
itemp = m;
*itemp = &c;
```

6. Write a program in C to find the factorial of a given number using pointers.**7. a) What are the values of main function variables x and y at the point marked / * values here */ in the following program?**

```
/* nonsense */
void silly(int x);
int
main(void)
{
    int x, y;
    x = 10; y = 11;
    silly(x);
    silly(y);                /* values here */
    . . .
}
void
silly(int x)
{
    int y;
    y = x + 2;
    x *= 2;
}
```

b)

```
/* nonsense */
void silly(int *x);
int
main(void)
{
    int x, y;
    x = 10; y = 11;
    silly(&x);
    silly(&y); /* values here */
    . . .
}
void
silly(int *x)
```

```
{
    int y;
    y = *x + 2;
    *x = 2 * *x;
}
```

8. What is the output of the following programs?

```
#include <stdio.h>

int main (void)
{
    int    count = 10, x;
    int    *int_pointer;

    int_pointer = &count;
    x = *int_pointer;

    printf ("count = %i, x = %i\n", count, x);

    return 0;
}
```

```
#include <stdio.h>

int main (void)
{
    char c = 'Q';
    char *char_pointer = &c;

    printf ("%c %c\n", c, *char_pointer);

    c = '/';
    printf ("%c %c\n", c, *char_pointer);

    *char_pointer = '(';
    printf ("%c %c\n", c, *char_pointer);

    return 0;
}
```

9. Write a statement to execute the following:

- Declare a pointer to an integer called `address`.
- Assign the address of a float variable `balance` to the float pointer `temp`.
- Assign the character value 'w' to the variable pointed to by the char pointer `letter`.
- Declare a pointer to the text string "Hello" called `message`.
- Assume float `balance [10] [5]`. How can you access `balance [3] [1]` using pointers?

10. Allocate memory to store an array of 50 integers and initialize elements of this dynamic array with values equal to their indexes (0 – 49). The code must provide memory allocation error checking.

11. Assuming that the function `getNextValue()` correctly returns a `float` type value, what is wrong with this code and fix it?

```
int i;
float *flArr;

if( (flArr = calloc(256, sizeof(float))) == NULL)
{
    fprintf(stderr, " Memory allocation failure"); return
    (-1);
}

for(i=0; i<256; i++)
    *flArr++ = getNextValue();

free( flArr );
```



12. Define a function

```
char *createEmptyString( int length);
```

that creates an empty c-string of a specified length. Write a simple `main()` function to test `createEmptyString()`.

13. Write a sample of code to allocate memory for:

- one item of type `component` and initialize it
- A component consists of `char type[7]`; `float price`; `int quantity`;
- a dynamic array with `SIZE` elements of type `component` and initialize the 3rd element.

14. Differentiate between dynamic and nondynamic data structures.