28/8/18

Assignment 1

☆ Write the features of C language.

-> C is a high-level, structure oriented programming language, used in general-purpose programming. It was developed by Dennis Ritchie at AT&T Bell Laboratories, USA between 1969 and 1973. Some of its features are as follows:

1. C is a robust language with a rich set of built-in functions and operators that can be used to write any complex program.
2. The C compiler combines the capabilities of an assembly-level language with features of a high-level language.
3. Programs written in C are efficient and fast due to its variety of data types and powerful operators.
4. C is highly portable i.e. programs once written can be run on another machines with little or no modification.
5. Another important feature of a C program, is its ability to extend itself.
6. A program in C is basically a collection of functions that are supported by the C Library. We can also create our own functions and add it to the C Library.
7. Nowadays, C is one of the most used languages for operating systems and embedded systems development.

✡ Write about any 5 Integrated Development Environments in which we can C programs.

→ i) Code :: Blocks - Code :: Blocks is an open source, cross platform and extensible IDE for C & C++. The best feature of this IDE is that it can be extended as per your requirements with the help of available plugins.

ii) Dev C++ - Dev C++ makes use of Min GW port of GCC as its compiler. Dev C++ also supports C language. Its features include the GCC based compiler, auto code completion, syntax highlighting, project manager and print support.

iii) Eclipse - Eclipse IDE is an open source utility that offers some advanced functionalities for C/C++ programmers. It has some impressive features such as syntax highlighting, auto code completion and an inbuilt debugger. In addition, Eclipse IDE makes coding simpler for new programmers.

iv) Sky IDE - Sky IDE is a multi-compiler, multi-view, multi-project and multi-profile C/C++ IDE. It also supports various other languages such as Java, PHP and Java Script. In addition, Sky IDE also has powerful text manipulation, syntax highlighting, auto code completion and line tracker functions.

v) NetBeans IDE - NetBeans is another advance open source IDE with features such as semantic highlighting, automatic formatting, auto code completion, unit testing, code assistance and much more.

✡ Describe the format specifiers and escape sequences used in the C language with the help of examples.

→ Format Specifiers can be defined as the operators which are used in association with the printf() function for printing the data that is referred by any object or any variable. When a value is stored in a particular variable, you cannot print the stored value straightforwardly without using the format specifiers. Format Specifiers start with a modulus '%' operator, followed by a special character for identifying the data type.

There are mainly 6 types of format specifiers available in C. They are as follows:

i) Integer Format Specifier (%d) - The %d format specifier is implemented for representing integer values. For example:

```
int x = 10;
printf("%d", x);
```

ii) Float Format Specifier (%f) - The %f format specifier is implemented for representing fractional values. For example:

```
float x = 14.5;
printf("%f", x);
```

DATE: / /

iii) Character Format Specifier (%.c) - The %.c format specifier is implemented for representing characters. For examples:

```
char x = 'f';
printf("%.c", x);
```

iv) String Format Specifier (%.s) - The %.s format specifier is implemented for representing strings.

v) Unsigned Integer Format Specifier (%.u) - The %.u format specifier is implemented for fetching values from the address of a variable having unsigned decimal integer stored in the memory.

vi) Long Int Format Specifier (%.ld) - The %.ld format specifier is implemented for representing long integer values.

→ C supports some character constants having a back slash (\) in front of them. These backslash character constants have a specific meaning which is known to the compiler. They are also termed as an 'Escape Sequence'. Some of them are as follows:

i) \a - Alert or Bell
ii) \b - Backspace
iii) \n - New line
iv) \r - Carriage return
v) \t - Horizontal tab

✡ Write the precedence of operators in C.

→ The operators with the highest precedence appear at the top of the table, those with the lowest appear at the bottom. Within an expression, the higher precedence operators will be evaluated first.

Category	Operators	Associativity
Postfix	() [] -> . ++ --	Left to Right
Unary	+ - ! ~ ++ -- (type)* & sizeof	Right to Left
Multiplicative	* / %	Left to Right
Additive	+ -	Left to Right
Shift	<< >>	Left to Right
Relational	< <= > >=	Left to Right
Equality	== !=	Left to Right
Bitwise AND	&	Left to Right
Bitwise XOR	^	Left to Right
Bitwise OR	|	Left to Right
Logical AND	&&	Left to Right
Logical OR	||	Left to Right
Conditional	? :	Right to Left
Assignment	= += -= *= /= %= >>= <<= &= ^= |=	Right to Left
Comma	,	Left to Right

12/10/18

Assignment 2

☆ Differentiate between 'while' and 'do... while' loops.

→

while	do... while
In while loop, the controlling condition appears at the start of the loop.	In do... while loop, the controlling condition appears at the end of the loop.
The iterations do not occur if the condition at the first iteration appears false.	The iterations occurs at least once even if the condition is false at the first iteration.

Syntax:
while (condition) {
statements ;
}
	Syntax:
	do {
	statements ;
	} while (condition);

☆ Write a note on 'continue' statement.

→ The 'continue' statement in C works somewhat like the break statement. Instead of forcing termination, it forces the next iteration of the loop to take place, skipping any code in between.

For the 'for' loop, continue statement causes the conditional test and increment portions of the loop to execute. For the 'while' and 'do... while' loops, continue statement causes the program to pass to the conditional tests.

✡ Write a program to print the following pattern :-

```
    *
   **
  ***
 ****
*****
```

```
#include <stdio.h>
#include <conio.h>

void main () {
  int i, j, k;
  for(i=1; i<=5; i++){
    for(j=4; j>=i; j--){
      printf(" ");
    }
    for(k=1; k<=i; k++){
      printf("*");
    }
    printf("\n");
  }
  getch();
}
```

✡ Write a program to calculate the sum of the following series:

$$2^3 - 4^3 + 6^3 - 8^3 + 10^3 \ldots n^3$$

```
#include <stdio.h>
#include <conio.h>

void main()
{
    int i, n, sum = 0, sign = 1;
    printf("Enter a number ");
    scanf("%d", &n);
    for(i = 2; i <= n; i += 2) {
        sign = (-1) * sign;
        sum = sum + sign * i * i * i;
    }
    printf("Sum of Series is %d", sum);
    getch();
}
```

Assignment 3

✡ Write a program to calculate the sum of digits of a number using function.

→
```
#include <stdio.h>
#include <conio.h>

int sum ( ) {
  int n, t, rem, sum = 0;
  printf (" Enter a number ");
  scanf (" %d ", &n);
  t = n;
  while ( t != 0) {
    rem = t %.10;
    sum = sum + rem
    t = t /10;
  }
  return ( sum );
}

void main ( ) {
  int sum (void);
  int s;
  s = sum ( );
  printf (" Sum of digits is %d ", s);
  getch ( );
}
```

✡ Write a note on Local & Global Variables with example.

→ Local Variables - Variables that are declared inside a block or function are called local variables. They can only be used by statements that are inside that block or function. They are not known to functions outside their own.

Global Variables - Global variables are defined outside a function, usually at the beginning of a program. They hold their value throughout the lifetime of the program. and they can be accessed inside any of the functions in the program.

Example :

```
#include <stdio.h>
#include <conio.h>

int g;  // global variable
void main ()
{
  int a, b;  // local variables
  a = 10;  b = 20;
  g = a+b;
  printf ("Sum is %d", g);
  getch ();
}
```

* What are Pointers ? Crive Example.

→ A pointer is a variable whose value is the address of another variable i.e., direct address of the memory location. Like any variable or constant, you must declare a pointer before using it to store any variable's address. For example :

```
#include <stdio.h>
#include <conio.h>

void main (){
  int var = 20;
  int *ip;                    // * - value at
  ip = &var;                  // & - address of

  printf (" Address of var variable - %x\n", &var);

  printf ("Address stored in ip variable - %x\n", ip);

  printf (" Value of *ip variable - %x\n", *ip);

  getch ();
}
```

✡ Write a program to reverse a given number using function.

→
```
#include <stdio.h>
#include <conio.h>

int reverse (){
  int n, rem, revno = 0;
  printf ("Enter a number with 3 or more digits ");
  scanf ("%d", &n);
  while (n != 0){
    rem = n % 10;
    revno = revno * 10 + rem;
    n /= 10;
  }
  return (revno);
}

void main (){
  int reverse (void);
  int rev;
  rev = reverse ();
  printf ("Reversed number is %d", rev);
  getch ();
}
```

14/11/18

PAGE NO.: 13
DATE: / /

Assignment 4

✡ Write a program to search the location of a given number in an array.

→
```
#include <stdio.h>
#include <conio.h>

void main () {
  int a[5], ele, i;
  printf ("Enter 5 values ");
  for( i = 0; i < 5; i++) {
     scanf ("%d", &a[i]);
  }
  printf ("Enter the number to be searched ");
  scanf ("%d", &ele);
  i = 0;
  while ( i < 5 && ele != a[i]) {
    i++;
  }
  if (i < 5)
    printf ("Number found at location = %d", i+1);
  else
    printf ("Number not found ");
  getch ();
}
```

Teacher's Signature.....

* Write a program to sort 10 numbers using Bubble sort method

→

```
#include <stdio.h>
#include <conio.h>

void main() {
  int array[10], n=10, c, d, swap;
  printf("Enter 10 numbers ");
  for(c=0; c<n; c++){
    scanf("%d", &array[c]);
  }

  for(c=0; c<n-1; c++){
    for(d=0; d<n-c-1; d++){
      if(array[d] > array[d+1]){
        swap = array[d];
        array[d] = array[d+1];
        array[d+1] = swap;
      }
    }
  }

printf("Sorted list in ascending order: \n");
  for(c=0; c<n; c++){
    printf("%d\n", array[c]);
  }
  getch();
}
```

✡ What is a 2-d Array ? Crive example

→ An array of arrays is known as 2D array. It is also known as a matrix. A matrix can be represented as a table with rows and columns. A 2-dimensional array can be considered as a table which will have x no. of rows and y number of columns. For example, a 2-D array 'a', which contains 3 rows and 4 columns can be shown as follows :

	Column 0	Column 1	Column 2	Column 3
Row 0	a [0] [0]	a [0] [1]	a [0] [2]	a [0] [3]
Row 1	a [1] [0]	a [1] [1]	a [1] [2]	a [1] [3]
Row 2	a [2] [0]	a [2] [1]	a [2] [2]	a [2] [3]

Thus, every element in the array 'a' is identified by an element name of the form a [i] [j], where 'a' is the name of the array and 'i' & 'j' are the subscripts that uniquely identify each element in 'a'

✡ Write a program to subtract 2 matrixes.

→
```
#include <stdio.h>
#include <conio.h>

void main ()
{
   int m, n, c, d, first [10] [10], second [10] [10], diff [10] [10];
   printf ("Enter no. of rows & columns of matrix \n");
   scanf ("%d %d", &m, &n);
```

PTO →

Teacher's Signature.....

```
printf("Enter the elements of first matrix\n");
for(c=0; c<m; c++){
    for(d=0; d<n; d++){
        scanf("%d", &first[c][d]);
    }
}

printf("Enter the elements of second matrix\n");
for(c=0; c<m; c++){
    for(d=0; d<n; d++){
        scanf("%d", &second[c][d]);
    }
}

printf("Difference of entered matrix :\n");
for(c=0; c<m; c++){
    for(d=0; d<n; d++){
        diff[c][d] = first[c][d] - second[c][d];
        printf("%d\t", diff[c][d]);
    }
    printf("\n");
}
getch();
}
```