Lecture 4, 30 July 2018

Decision making

```
int main() {
   int n1, n2; /* 2 integers are declared */
   printf("Give 2 integers: ");
   scanf("%d%d", &n1, &n2);
   if( n1 == n2 ) printf("%d and %d are equal.\n", n1, n2);
   if( n1 != n2 ) printf("%d and %d are not equal.\n", n1, n2);
   if( n1 < n2 ) printf("%d is smaller than %d\n", n1, n2);
   if( n1 > n2 ) printf("%d is bigger than %d\n", n1, n2);
   if( n1 <= n2 ) printf(""%d is smaller than or equal to %d\n", n1, n2);
   if( n1 >= n2 ) printf("%d is bigger than or equal to %d\n", n1, n2);
   return 0;
}
```

We will read values of `n1` and `n2` while execution. All values of `n1` and `n2` will fulfil 3 conditions in above program. In in output, we will get three outputs.

Decision making (with AND logic)

```c
int main() {
   float grade;
   printf("Give your grade: ");
   scanf("%f", &grade);
   if( grade >= 90 ) printf("Your grade is O (outstanding).\n");
   if( grade < 90 && grade >= 80 ) printf("Your grade is A (Very good).\n");
   if( grade < 80 && grade >= 70 ) printf("Your grade is B (Good).\n");
   if( grade < 70 && grade >= 60 ) printf("Your grade is C (Normal).\n");
   if( grade < 60 && grade >= 50 ) printf("Your grade is D (Bad).\n");
   if( grade < 50 ) printf("You are failed.\n");
   return 0;
}
```

Here we are using `AND logic` in above if-statements. In c-programming, we use `&&` (double ampersand) in two logics for `AND logic`. If we need `OR logic`, we use `||` (double pipe lines).

```
1 && 1 is 1 => Yes && Yes = Yes
0 || 1 is 1 => No || Yes = Yes
```

Equality relations operators

Algebraic equality	C equality	Example in C	Meaning
Equality Operators			
=	==	x == y	x is equal y
≠	!=	x != y	x is not equal y
Relational Operators			
>	>	x > y	x is greater than y
<	<	x < y	x is less than y
≥	>=	x >= y	x is greater than or equal y
≤	<=	x <= y	x is less than or equal y

Decision making (`if-else`)

```
int main() {
   float grade;
   printf("Give your grade: ");
   scanf("%f", &grade);
   if( grade >= 90 ) {
      printf("Your grade is O (outstanding).\n");
   } else {
      if( grade >= 80 ) {
         printf("Your grade is A (Very good).\n");
      } else {
         if( grade >= 70 ) {
            printf("Your grade is B (Good).\n");
         } else {
            if( grade >= 60 ) {
               printf("Your grade is C (Normal).\n");
            } else {
               if( grade >= 50 ) {
                  printf("Your grade is D (Bad).\n");
               } else {
                  printf("You are failed.\n");
   }}}}}
   return 0;
}
```

Repeated from previous slide (`if-else`)

```c
int main() {
   float grade;
   printf("Give your grade: ");
   scanf("%f", &grade);
   if( grade >= 90 ) {
      printf("Your grade is O (outstanding).\n");
   } else if( grade >= 80 ) {
      printf("Your grade is A (Very good).\n");
   } else if( grade >= 70 ) {
      printf("Your grade is B (Good).\n");
   } else if( grade >= 60 ) {
      printf("Your grade is C (Normal).\n");
   } else if( grade >= 50 ) {
      printf("Your grade is D (Bad).\n");
   } else {
      printf("You are failed.\n");
   }
   return 0;
}
```

Decision making (if-else)

```
int main() {
   char grade;
   printf("Give your grade: ");
   scanf("%c", &grade);
   if( grade == 'A' || grade == 'a' ) {
      printf("Very good.\n");
   } else if( grade == 'B' || grade == 'b' ) {
      printf("Good\n");
   } else if( grade == 'C' || grade == 'c' ) {
      printf("Normal\n");
   } else if( grade == 'D' || grade == 'd' ) {
      printf("Bad\n");
   } else if( grade == 'E' || grade == 'e' ) {
      printf("Very bad\n");
   } else if( grade == 'F' || grade == 'f' ) {
      printf("Fail\n");
   } else if( grade == '\n' || grade == '\t' || grade == ' ' ){
   } else {
      printf("Wrong grade entered.\n");
   }
   return 0;
}
```

if-else converted to witch-case

```
int main() {
    char grade;
    printf("Give your grade: ");
    scanf("%c", &grade);
    switch(grade){
    case 'A':
    case 'a':
       printf("Very good.\n");
       break;
    case 'B':
    case 'b':
       printf("Good\n");
       break;
    case 'C':
    case 'c':
       printf("Normal\n");
       break;
    case 'D':
    case 'd' :
       printf("Bad\n");
       break;
    case 'E':
    case 'e' :
       printf("Very bad\n");
       break;
    case 'F':
    case 'f' :
       printf("Fail\n");
       break;
    case '\n':
    case '\t':
    case ' ':
       break;
       // to ignore newline, tab and space
    default:
       printf("Wrong grade entered.\n");
    }
    return 0;
}
```

`for` Repetition Statement

```
int main() {
   int i, j;
   printf("Enter an integer: ");
   scanf("%d", &j);
   for( i = 1; i <= 10; i = i + 1 ){
      printf("%d\n", i * j );
   }
   return 0;
}
```

Above program will print a table of given integer. In this program, we are reading a variable stored in variable named j. While in for loop, we are printing that variable with multiplying i. Since, I gives the repetition to the loop and increasing its value.

Lecture 5, 31 July 2018

Few operations:

Assignment Operator	Sample Expression	Example	Assignment
Assume: `int a = 2, b = 9, c = 4, d = 27, h = 15, j = 10, k = 15, m = 28, n =24;`			
`+=`	`a += 6`	`a = a + 6`	Assign 8 to `a`
`-=`	`b -= 4`	`b = b - 4`	Assign 5 to `b`
`*=`	`c *= 3`	`c = c * 3`	Assign 12 to `c`
`/=`	`d /= 3`	`d = d / 3`	Assign 9 to `d`
`%=`	`e %= 4`	`e = e % 4`	Assign 3 to `e`
`++`	`j++` OR `++k`	`j=j+1` OR `k= k+1`	Assign 11 to `j` OR assign 16 to `k`
`--`	`j--` OR `--k`	`m=m-1` OR `n= m-1`	Assign 27 to `m` OR assign 23 to `n`

for Repetition Statement

```c
/*
  File with name lecture5_1.c
  Class average / counter-controlled repetition
*/
#include <stdio.h>
int main() {
   float counter, grade, total, average;
   total = 0;
   for( counter = 1; counter < 10; counter = counter + 1 ){
      printf("Enter grade: ");
      scanf("%d", &grade);
      total = total + grade;
   }
   average = total / 10;
   printf("class average = %d\n", average);
   return 0;
}
```

for Repetition Statement (another way, not good)

```
/*
  File with name lecture5_2.c
  Class average / counter-controlled repetition
*/
#include <stdio.h>
int main() {
   float counter = 1, grade, total, average;
   total = 0;
   for(  ; counter < 10; ){
      printf("Enter grade: ");
      scanf("%d", &grade);
      total = total + grade;
      counter = counter + 1;
   }
   average = total / 10;
   printf("class average = %d\n", average);
   return 0;
}
```

for Repetition Statement with different condition

```c
/*    File name: lecture5_3.c
      counter-controlled repetition  */
#include <stdio.h>
int main () {
   int counter , grade, total = 0;
   float average;
   printf("Enter grade, -1 to end: ");
   scanf("%d", &grade);
   for( counter = 0; grade != -1; counter = counter + 1 ){
      total = total + grade;
      printf("Enter grade, -1 to end: ");
      scanf("%d", &grade);
   }
   if( counter != 0 ) {
      average = (float) total / counter;
      printf("Class average is %.3f\n", average);
   } else printf("No grades were entered.\n");
   return 0;
}
```

for Repetition Statement (repeated from last page)

```
/*    File name: lecture5_4.c
      counter-controlled repetition  */
#include <stdio.h>
int main () {
   int counter = 0, grade, total = 0;
   float average;
   printf("Enter grade, -1 to end: ");
   scanf("%d", &grade);
   for(  ; grade != -1; ){
      total = total + grade;
      counter = counter + 1;
      printf("Enter grade, -1 to end: ");
      scanf("%d", &grade);
   }
   if( counter != 0 ) {
      average = (float) total / counter;
      printf("Class average is %.3f\n", average);
   } else printf("No grades were entered.\n");
   return 0;
}
```

Change for in while

```
/*    File name: lecture5_5.c
      counter-controlled repetition  */
#include <stdio.h>
int main () {
   int counter = 0, grade, total = 0;
   float average;
   printf("Enter grade, -1 to end: ");
   scanf("%d", &grade);
   while(  grade != -1 ){
      total = total + grade;
      counter = counter + 1;
      printf("Enter grade, -1 to end: ");
      scanf("%d", &grade);
   }
   if( counter != 0 ) {
      average = (float) total / counter;
      printf("Class average is %.3f\n", average);
   } else printf("No grades were entered.\n");
   return 0;
}
```

Change `while` in `do-while`

```c
/*    File name: lecture5_6.c
      counter-controlled repetition  */
#include <stdio.h>
int main () {
   int counter = 0, grade, total = 0;
   float average;
   do {
      printf("Enter grade, -1 to end: ");
      scanf("%d", &grade);
      total = total + grade;
      counter = counter + 1;
   } while(  grade != -1 );
   total -= 1;
   if( counter != 0 ) {
      average = (float) total / counter;
      printf("Class average is %.3f\n", average);
   } else printf("No grades were entered.\n");
   return 0;
}
```

Lecture 6, 1 August 2018

do while Repetition Statement

```c
/*
   File name: lecture6_1.c
   counter-controlled (do while) repetition
*/
#include <stdio.h>
int main () {
   int counter = 1;
   do {
      printf("%d ", counter);
   } while( ++counter <= 10 );
   printf("\n");
   return 0;
}
```

`break` and `continue` Statement in loop

```c
/* File name: lecture6_2.c
   break and continue statements in for loop */
#include <stdio.h>
int main () {
   int i, ii;
   printf("Break statement in first loop\n");
   for( i = 1; i <= 10; i = i + 1 ){
      if( i == 5 ) break;
      printf("%d ", i);
   }
   printf("\nLoop is broken at i = %d\n\n", i);
   for( ii = 1; ii <= 10; ii = ii + 1 ){
      if( i == 5 ) continue;
      printf("%d ", ii);
   }
   printf("\n"); return 0;
}
```

switch-case Statement (1)

```
/*
   File name: lecture6_3.c
   Learning simple switch-case in c-language
*/
#include <stdio.h>
int main () {
   char x;
   float y1, y2;
   printf("Enter operator either + or - or * or divide : ");
   scanf("%c", &x);
   printf("Enter two operands (real numbers): ");
   scanf("%f%f", &y1, &y2);
   switch( x ){
      case '+' :
         printf("%.3f + %.3f = %.3f\n", y1, y2, y1 + y2);
         break;
      case '-' :
         printf("%.3f + %.3f = %.3f\n", y1, y2, y1 - y2);
```

switch-case Statement (1)

```
        case '-' :
            printf("%.3f + %.3f = %.3f\n", y1, y2, y1 - y2);
            break;
        case '*' :
            printf("%.3f + %.3f = %.3f\n", y1, y2, y1 * y2);
            break;
        case '/' :
            printf("%.3f + %.3f = %.3f\n", y1, y2, y1 / y2);
            break;
        default :
            printf("Error! operator is not correct\n");
            break;
    }
    return 0;
}
```

switch-case Statement (2)

```
/*
   File name: lecture6_4.c
   Learning simple switch-case in c-language
*/
#include <stdio.h>
int main () {
   int grade, aCount, bCount, cCount, dCount, eCount, fCount;
   aCount = bCount = cCount = dCount = eCount = fCount=0;
   printf("Enter the letter grades.\n");
   printf("Enter EOF character to end the input.\n");
   while( (grade = getchar()) != EOF ) { /*  press Ctrl+D for EOF */
      switch( grade ){
          case 'A' :
          case 'a' :
             aCount++;
             break;
          case 'B' :
          case 'b' :
```

switch-case Statement (2)

```
            case 'b' :
               bCount++;
               break;
            case 'C' :
            case 'c' :
               cCount++;
               break;
            case 'D' :
            case 'd' :
               dCount++;
               break;
            case 'E' :
            case 'e' :
               eCount++;
               break;
            case 'F' :
            case 'f' :
               fCount++;
               break;
            case '\n' :
            case '\t' :
            case ' ' :
               break;
            default :
               printf("Incorrect grade entered.");
               printf("Enter new grade.\n");
               break;
         }
      }
      printf("\nTotals for each letter grade are:\n" );
      printf("A: %d\n", aCount );
      printf("B: %d\n", bCount );
      printf("C: %d\n", cCount );
      printf("D: %d\n", dCount );
      printf("E: %d\n", eCount );
      printf("F: %d\n", fCount );
      return 0;
}
```

Lecture 7, 2 August 2018

Math library functions

```c
/*
   File name: lecture7_1.c
   Math library and it's functions
*/
#include <stdio.h>
#include <math.h>
int main () {
   float a=0.5, b = -0.5;
   printf("square root of %f = %f\n", a, sqrt(a));
   printf("exponential of %f = %f\n", a, exp(a));
   printf("exponential of %f = %f\n", b, exp(b));
   return 0;
}
```

To compile this program we should use following command:

```
$ gcc   -o    <file name>    <file name>.c    -lm
```

Few math functions and their meaning

- `sqrt(x)` :– Square root of given number x. $\forall\ x \geq 0$.
- `exp(x)` :– Exponential of x. $\forall\ \ x \in R$
- `log(x)` :– Natural logarithm of x. $\forall\ x > 0$.
- `log10(x)` :– Logarithm of x on base 10. $\forall\ x \geq 0$.
- `fabs(x)` :– Absolute value of x. $\forall\ x \in R$.
- `ceil(x)` :– Smallest integer not less than x. $\forall\ x \in R$.
- `floor(x)` :– Largest integer not greater than x. $\forall\ x \in R$.
- `pow(x, y)` :– x raised to the power y. $\forall\ x, y \in R$.
- `fmod(x, y)` :– Remainder of x/y as a floating-point number. $\forall\ x, y \in R$.
- `sin(x)` :– Trigonometric sine of x. $\forall\ x \in R$.
- `cos(x)` :– Trigonometric cosine of x. $\forall\ x \in R$.
- `tan(x)` :– Trigonometric tangent of x. $\forall\ x \in R$.
- `asin(x)` :– Trigonometric sine inverse of x. $\forall\ -1 \leq x \leq 1$

- `asin(x)` :– Trigonometric sine inverse of x. ∀ − 1 ≤ x ≤ 1
- `acos(x)` :– Trigonometric cosine inverse of x. ∀ − 1 ≤ x ≤ 1
- `atan(x)` :– Trigonometric tangent inverse of x. ∀ − 1 ≤ x ≤ 1
- `sinh(x)` :– Hyperbolic sine of x. ∀ x ∈ R.
- `cosh(x)` :– Hyperbolic cosine of x. ∀ x ∈ R.
- `tanh(x)` :– Hyperbolic tangent of x. ∀ x ∈ R.
- `asinh(x)` :– Hyperbolic sine inverse of x. ∀ x ∈ R.
- `acosh(x)` :– Hyperbolic cosine inverse of x. ∀ x ∈ R+.
- `atanh(x)` :– Hyperbolic tangent inverse of x. ∀ − 1 ≤ x ≤ 1
- `exp2(x)` :– 2 raised to the power x. ∀ x ∈ R.
- `exp2f(x)` :– 2 raised to the power x as floating number. ∀ x ∈ R.
- `expm1(x)` :– exp(x) - 1. ∀ x ∈ R.
- `expm1f(x)` :– exp(x) - 1. ∀ x ∈ R.
- `exp2l(x)` :– 2 raised to the power x as `long double`. ∀ x ∈ R.
- `expm1l(x)` :– exp2l(x) - 1. ∀ x ∈ R.

Programmer defined function (1)

```
/*
   File name: lecture7_2.c
   creating and using a programmer-defined function
*/
#include <stdio.h>
int square (int); /* function prototype */
int main () {
   int i;
   for ( i = 1; i <= 10; i = i + 1 ){
      printf("%d ", square(i));
   }
   printf("\n");
   return 0;
}
int square (int a){return a*a;}
```

Programmer defined function (2)

```c
#include <stdio.h>
int maximum (int, int, int);
int minimum (int, int, int);
int main () {
   int n1, n2, n3;
   printf("Enter 3 integers: ");
   scanf("%d%d%d", &n1, &n2, &n3 );
   printf("Maximum is: %d\nand Minumn is: %d\n",
          maximum(n1,n2,n3), minimum(n1,n2,n3));
   return 0;
}
int maximum (int x, int y, int z){
  int max=x;
  if( y > max ) max=y;
  if( z > max ) max=z;
  return max;
}
```

```c
int minimum (int a, int b, int c){
  int min=a;
  if( b < min ) min=b;
  if( c < min ) min=c;
  return min;
}
```