CS 2211
Systems Programming

Part Three: Basic Control in C

CONTROL in C

Three Control Statements in C:

    SELECTION Statements:
      IF statements and SWITCH statements

    ITERATION Statements (loops):
      WHILE loops
      DO loops
      FOR loops

    JUMP Statements:
      BREAK
      CONTINUE
      GOTO
      RETURN

other C statements:
  COMPOUND statements
  NULL statements

IF statements

```c
#include <stdio.h>

int main(int argc, char *argv[])
{
    int age;                                    /* Need a variable... */

    printf( "Please enter your age" );          /* Asks for age */
    scanf( "%d", &age );                        /* The input is put in age */
    if ( age < 10 ) {                           /* If age is less than 100 */
      printf ("You are an adolescent.\n" );
    }
    else if ( age > 12 && age < 20 )
      printf( "You are old\n" );      /* bracket optional if one line */

    else
    {                        /* bracket one separate (stand-alone) line */
      printf( "You are no longer the target market\n" );
    }

    return 0;
}
```

IF statements

```c
#include <stdio.h>

int main(int argc, char *argv[])
{
    int age;

    printf( "Please enter your age" );
    scanf( "%d", &age );
    if ( age < 10 ) {
      printf ("You are an adolescent.\n" );
    }
    else if ( age > 12 && age < 20 )
      printf( "You are old\n" );      /* bracket optional if one line */

    else
    {                    /* bracket one separate (stand-alone) line */
      printf( "You are no longer the target market\n" );
    }

    return 0;
}
```

!	logical negation
&&	logical and
||	logical or
0	false
1	(or anything else) true
==	equal to
!=	not equal to

MULTIPLE IF statements

```c
#include <stdio.h>

int main(int argc, char *argv[])
{
    int grade;                              /* Need a variable... */

        if (grade == 3)
          printf("Good");
        else if (grade == 2)
          printf("Average");
        else if (grade == 1)
          printf("Poor");
        else if (grade == 0)
          printf("Failing");
        else
          printf("Illegal grade");

    return 0;
}
```
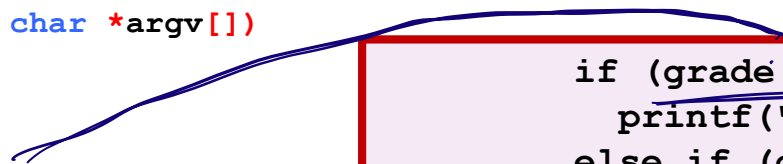
SWITCH statements

```c
#include <stdio.h>

int main(int argc, char *argv[])
{
    int grade;

        switch (grade) {
          case 3:
            printf("Good");
             break;
          case 2:
            printf("Average");
             break;
          case 1:
            printf("Poor");
              break;
          case 0:
             printf("Failing");
               break;
          default:
             printf("Illegal grade");
              break;

    return 0;
}
```

```c
            if (grade == 3)
              printf("Good");
           else if (grade == 2)
              printf("Average");
           else if (grade == 1)
              printf("Poor");
           else if (grade == 0)
              printf("Failing");
           else
              printf("Illegal grade");
```

SWITCH statements

```c
#include <stdio.h>

int main(int argc, char *argv[])
{
    int grade;                              /* Need a variable... */

        switch (grade) {
          case 3:  printf("Good");          // Without break (or some
          case 2:  printf("Average");       // other jump statement) at
          case 1:  printf("Poor");          // the end of a case,
          case 0:  printf("Failing");       // control will flow into
          default: printf("Illegal grade"); // the next case.
        }

    return 0;
}
```

C Basic Controls

END OF PART 1

WHILE loop

```c
#include <stdio.h>

int main (int argc, char *argv[])
{
    /* local variable definition */
    int a = 10;

    /* while loop execution */
    while( a < 20 )
    {
        printf("value of a: %d\n", a);
// a++;                                // !!!! INFINITE LOOP !!!!
    }

    return 0;
}
```

WHILE loop

```c
#include <stdio.h>

int main (int argc, char *argv[])
{
    int n = 0;

     while (n < 10)
     {
         n++;
         if (n % 2 == 1)                          /* check that n is odd */
         {
             continue;        /* go back to the start of the while block */
         }

         printf("The number %d is even.\n", n);     /* only if n is even */
     }

    return 0;
}
```

DO - WHILE loop

```c
#include <stdio.h>

int main (int argc, char *argv[])
{
   int j=0;

   do
   {
       printf("Value of variable j is: %d\n", j);
       j++;
   } while (j<=3);

   return 0;
}
```

> **WHILE** loops
>   PRE-TEST
>
> **DO-WHILE** loops
>   POST TEST

C Basic Controls

END OF PART 2

FOR loop

```
for (initializationStatement; testExpression; updateStatement)
{
    // statements inside the body of loop
}
```

```c
#include <stdio.h>

int main (int argc, char *argv[])
{

    int i;

    for (i = 1; i < 11; i++)
    {
        printf("%d \n", i);
    }

    return 0;
}
```

FOR loop

```c
#include <stdio.h>

int main (int argc, char *argv[])
{

    int i, j = 20, k = 40;

    for (i = 1; i < 11; i++, j--, k-4)
    {
        printf("i :%d and j: %d and k: %d", i, j, k);
    }

    return 0;
}
```

FOR loop

```c
#include <stdio.h>

int main (int argc, char *argv[])
{

    // int i, j = 20, k = 40;   // if only used WITHIN the loop

    for (int i = 1, j = 20, k = 40; i < 11; i++, j--, k-=4)
    {
        printf("i :%d and j: %d and k: %d", i, j, k);
    }

    return 0;
}
```

FOR loop

```c
#include <stdio.h>

int main (int argc, char *argv[])
{

 int i, k, levels, space;
    printf("Enter the number of levels in pyramid:");
    scanf("%d",&levels);

    space = levels;

    for ( i = 1 ; i <= levels ; i++ )
    {
        for ( k = 1 ; k < space ; k++ )
            printf(" ");
        space--;

        for ( k = 1 ; k <= 2*i - 1 ; k++ )
            printf("*");
        printf("\n");
    }

    return 0;
}
```

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

FOR loop with a BREAK

```c
#include <stdio.h>

int main (int argc, char *argv[])
{

    // Loops that read user input up to ten times,
    // terminating when a particular value
    // is entered, often fall into this category

   for (int i = 1; i<=10; i++)
   {
       printf("Enter a number (enter 0 to stop): ");
       scanf("%d", &n);

       if (n == 0)
          break;
        printf("%d cubed is %d\n", n, n * n * n);
   }

   return 0;
}
```

FOR loop with a NULL

```c
#include <stdio.h>

int main (int argc, char *argv[])
{

    char string = "Something or Another");
    {
        int i ;    // i MUST be declared OUTSIDE the loop to survive ...
        for ( i = 0 ; string [ i ] != '\0' ; i++ )
            /* NULL STATEMENT! */ ;

        return i ; // … otherwise i is dereferenced before control passes back
    }

    return 0;
}
```

C Basic Controls

END OF PART 3

CONDITIONAL OPERATOR (? symbol)

```c
#include<stdio.h>

int main()
{
int num;

printf("Enter the Number : ");
scanf("%d",&num);

flag = ((num%2==0)?1:0);

if(flag==0)
    printf("\nEven");
else
    printf("\nOdd");
}
```
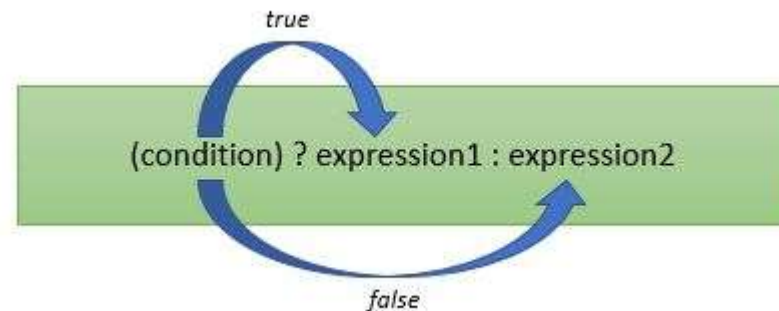
```c
#include<stdio.h>

int main()
{
int num;

printf("Enter the Number : ");
scanf("%d",&num);

(num%2==0)? printf("Even"): printf("Odd");

}
```

true

(condition) ? expression1 : expression2

false

CONDITIONAL OPERATOR (? symbol)

```c
#include<stdio.h>

int main()
{
double x, y, z;

x = 3;
y = 12;


if( x > y )
    z = x;
  else
    z = y;
}
```
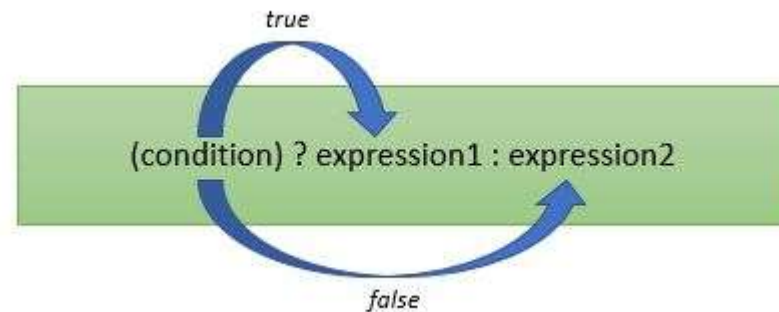
```c
#include<stdio.h>

int main()
{
double x, y, z;

x = 3;
y = 12;

z = ( x > y ) ? x : y;

}
```


true

(condition) ? expression1 : expression2

false

CONDITIONAL statements

```c
#include <stdio.h>

int main(int argc, char *argv[])
{
    int i, k;                            /* Need a variable... */

    i = 2;
    j = 3;

    k = i > j ? i : j;          /* k is now 2 */
    k = (i >= 0 ? i : 0) + j;   /* k is now 3 */


    return i > k ? i : k;
}
```

C Basic Controls

END OF PART 4