C Programming Introduction

Week 8:Loops

Topic of this week

- Loops
  - Class Lecture Review
    - The While,do Repetition Structure
    - Notes and Observations
    - Continue and break
  - Programming Exercises

The While,do Repetition Structure

- While Statement
  - The expression is evaluated. If it is *true*, statement is executed and expression is reevaluated. This cycle continues until expression becomes *false*.

```
while (expression) {
   Statement1;
   Statement2;
    ...
}
```

The While,do Repetition Structure

- Example of While

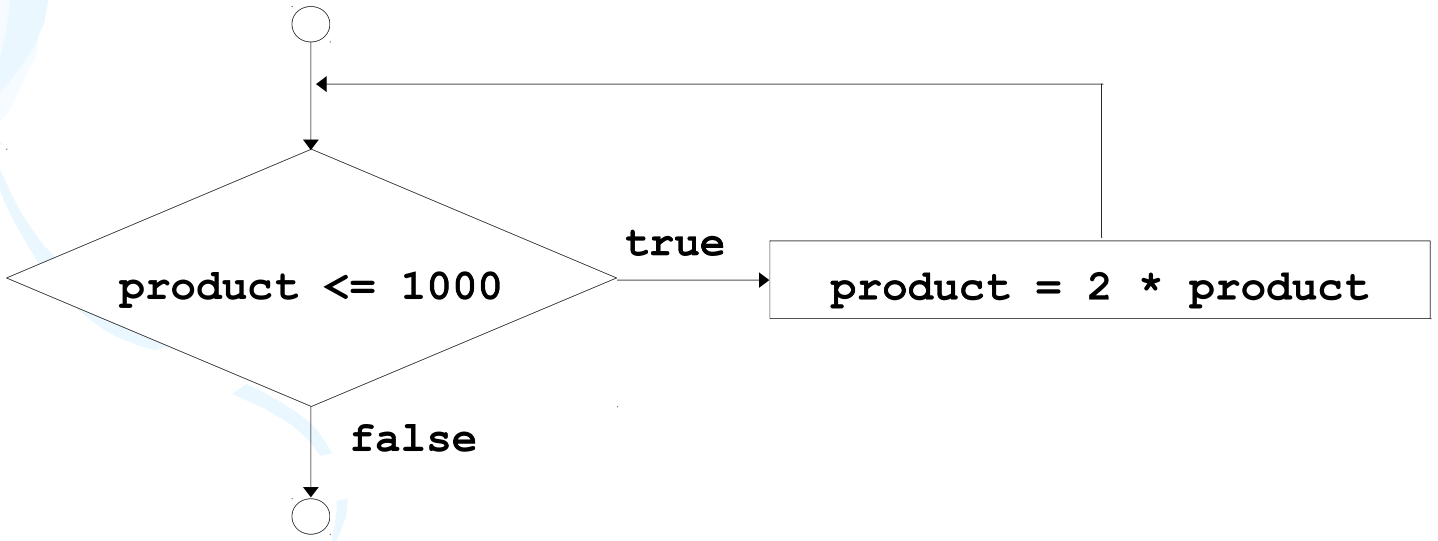
```
#include <stdio.h>
#define PERIOD ‘.’
main() {
   char C;
   while ((C = getchar())!= PERIOD)
      putchar(C);
   printf(“Good Bye.\n”);
}
```

Result?

The While,do Repetition Structure

- Example:

```
int product = 2;
while ( product <= 1000 )
   product = 2 * product;
```



The While,do Repetition Structure

- Do-While Statement
  - The do-while, tests at the bottom after making each pass through the loop body; the body is always executed at least once.

```
do {
    statement1;
    statement2;
    …
} while (expression);
```

The While,do Repetition Structure

- Example of Do-While

```
int i = 1, sum = 0;
do {
   sum += i;
   i++;
} while (i <= 50);
printf("The sum of 1 to 50 is %d\n", sum);
```

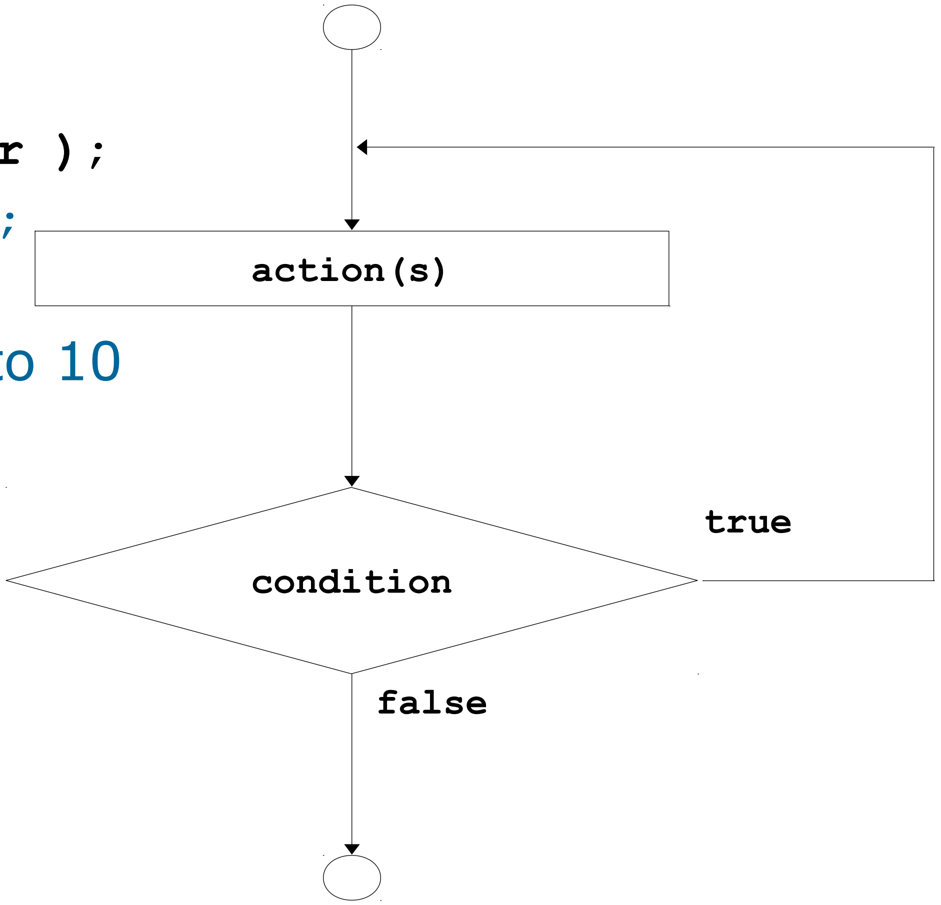
Result?

The While,do Repetition Structure

- Example (letting `counter = 1`)

```
do {
   printf( "%d  ", counter );
} while (++counter <= 10);
```

Prints the integers from 1 to 10



Continue and Break

- Break and Continue Statement
  - The break statement provides an early exit from for, while, and do.

```
break;
```

  - The continue statement is related to break, but less often used; it causes the next iteration of the enclosing for, while, or do loop to begin.

```
continue;
```

Continue and Break

- Example of Break and Continue

```
int c;
while ((c = getchar()) != -1) {
   if (C == '.')
      break;
   else if (c >= '0' && c <= '9')
      continue;
   else putchar(c);
}
printf("*** Good Bye ***\n");
```

Exercise 8.1

- Write a program that copies content inputed from the keyboard to the screen, but replace the sequence of blank characters by only one blank character.
- You can use getchar() and putchar() method to carry out this program.

Solution

```
#include <stdio.h>
int main(void)
{
  int c;
  int inspace;
  inspace = 0;
  while((c = getchar()) != EOF)
  {
    if(c == ' ')
    {
      if(inspace == 0)
      {
        inspace = 1;
        putchar(c);
      }
```

Solution

```
        }

        /* We haven't met 'else' yet, so we have to be a little clumsy */
        if(c != ' ')
        {
          inspace = 0;
          putchar(c);
        }
      }

      return 0;
    }
```

Exercise 8.2

- Write a program that replaces characters such as: tab,\t,\b by \\ character in the input string and print out.
- You can use getchar() method to carry out this program.
- You can use *if* structure or *switch* structure.

Solution

```
#include <stdio.h>

int main()
{
    int c, d;

    while ( (c=getchar()) != EOF) {
        d = 0;
        if (c == '\\') {
            putchar('\\');
            putchar('\\');
            d = 1;
        }
```

Solution

```
        if (c == '\t') {
            putchar('\\');
            putchar('t');
            d = 1;
        }
        if (c == '\b') {
            putchar('\\');
            putchar('b');
            d = 1;
        }
        if (d == 0)
            putchar(c);
    }
    return 0;
}
```

Exercise 8.3

- Calculate square cube by using newton method.

Solution

```
#include <stdio.h>
#include <math.h>
void main()
{
  double a, xn, ketqua;
  printf("\Enter the value need to be squared cube: ");
  scanf("%lf", &a);
  xn = (a+1)/2;
  do {
    ketqua = xn;
    xn = 0.5 * (xn + a/xn);
  } while (fabs(xn-ketqua) > 0.0001);
  printf("\nResult = %lf", xn);
}
```

Exercise 8.4

- How to compute the payroll for a company?
- Write and compile the program below to see how you can use while statement to do this task.

exercise8_4.c

```
#include <stdio.h>

int
main(void)
{
      double total_pay;    /* company payroll      */
      int    count_emp;    /* current employee     */
      int    number_emp;   /* number of employees */
      double hours;        /* hours worked         */
      double rate;         /* hourly rate          */
      double pay;          /* pay for this period */
      /* Get number of employees.                           */
      printf("Enter number of employees> ");
      scanf("%d", &number_emp);
/* Compute each employee's pay and add it to the payroll. */
    total_pay = 0.0;
    count_emp = 0;
    while (count_emp < number_emp) {
       printf("Hours> ");
       scanf("%lf", &hours);
       printf("Rate > $");
       scanf("%lf", &rate);
       pay = hours * rate;
       printf("Pay is $%6.2f\n\n", pay);
       total_pay = total_pay + pay;
       count_emp = count_emp + 1;
    }
     printf("All employees processed\n");
    printf("Total payroll is $%8.2f\n", total_pay);
    return (0);
}
```

Exercise 8.5

- Write a program that use *while* structure to analysis of examination results: how many passed students and failed students.
- You can simply ask user to show that a student is passed or failed by entering a presented number: 1 is passed and 2 is failed.

Solution

```
#include <stdio.h>

/* function main begins program execution */
int main( void )
{
   /* initialize variables in definitions */
   int passes = 0;   /* number of passes */
   int failures = 0; /* number of failures */
   int student = 1;  /* student counter */
   int result;       /* one exam result */

   /* process 10 students using counter-controlled loop */
   while ( student <= 10 ) {

      /* prompt user for input and obtain value from user */
      printf( "Enter result ( 1=pass,2=fail ): " );
      scanf( "%d", &result );
```

Solution

```
/* if result 1, increment passes */
    if ( result == 1 ) {
      passes = passes + 1;
    } /* end if */
    else { /* otherwise, increment failures */
      failures = failures + 1;
    } /* end else */

    student = student + 1; /* increment student counter */
  } /* end while */

  /* termination phase; display number of passes and failures */
  printf( "Passed %d\n", passes );
  printf( "Failed %d\n", failures );
```

Solution

```
/* if more than eight students passed, print "raise tuition" */
  if ( passes > 8 ) {
    printf( "Raise tuition\n" );
  } /* end if */

  return 0; /* indicate program ended successfully */

} /* end function main */
```

Exercise 8.6

- Use do…while statement to print out integers that is smaller than a preceded number.
- Note that the do…while statement always performs one time at least.

Solution

```
#include <stdio.h>

/* function main begins program execution */
int main( void )
{
   int counter = 1;              /* initialize counter */

   do {
      printf( "%d  ", counter ); /* display counter */
   } while ( ++counter <= 10 ); /* end do...while */

   return 0; /* indicate program ended successfully */

} /* end function main */
```

Exercise 8.7

- We would like a program to average a set of grades.
- Algorithm notes:
  - We need a running sum of grades, and a running count of how many grades have been read so far.
  - We need to read until we get a sentinel value | let's use a negative grade to indicate we are done.
  - Need to be sure we print prompts.

Solution using while

```
# include <stdio .h>
int main ()
{
    float grade , sum = 0.0;
    int gradeCount = 0;
    printf (" Enter grade : ");
    scanf ("%g", & grade );
    while ( grade >= 0.0) {
    sum += grade ;
    ++ gradeCount ;
    printf (" Enter grade : ");
    scanf ("%g", & grade );
    }
    printf (" Average : %g\n",
    sum/ gradeCount );
    return 0;
}
```

Solution using do...while

```
# include <stdio .h>
int main () {
    float grade , sum;
    int gradeCount ;
    int another ;
    do {
    sum = gradeCount = 0;
    printf (" Enter grade : ");
    scanf ("%g", & grade );
    while ( grade >= 0.0) {
    sum += grade ;
    ++ gradeCount ;
    printf (" Enter grade : ");
    scanf ("%g", & grade );
    }
    printf (" Average : %g\n\n",
    sum/ gradeCount );
    printf (" Another class : ");
    scanf ("%d", & another );
  } while ( another != 0);
  return 0;
}
```

Exercise 8.8

- Write a program that compute n! using a loop.
- You can use:
  - Counter" variable, i, ranging from 1 to n.
  - Running product f, tracking i!.

Solution

```
/* n! using while . */
# include <stdio .h>
int main () {
    int i, n, f;
    printf (" Enter n: ");
    scanf ("%d", &n);
    f = 1; /* 0! */
    i = 1;
    while (i <= n) {
    f *= i; /* Now , f = i! */
    ++i;
    }
    printf ("%d! = %d\n", n, f);
    return 0;
}
```