CSE 220
C Programming

Altering the Flow of a Loop
Announcement

• Project #2 will be posted by the end of this week.

• Homework #6 will be posted by the end of this week.

• Midterm exam has been graded.
Midterm Exam

- Minimum : 13.5
- Maximum : 53.5
- Average : 35.23
- Curve: ?
Midterm Exam

» Minimum : 13.5

» Maximum : 53.5

» Average : 35.23

» Curve: +5 to everyone’s grade
Example (I): Multiplication Table

• Write a program the given n, prints the multiplication table for n:

Input: 5
Output: a 5 by 5 multiplication table

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>8</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>6</td>
<td>9</td>
<td>12</td>
<td>15</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>8</td>
<td>12</td>
<td>16</td>
<td>20</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>10</td>
<td>15</td>
<td>20</td>
<td>25</td>
</tr>
</tbody>
</table>
Example (I): Multiplication Table
Example (I): Multiplication Table

```c
#include <stdio.h>

int main()
{
    int n;
    printf("Enter the number: ");
    scanf("%d", &n);

    for(int i = 1 ;i <= n; i++)
    {
        printf("\n");
    }
    return 0;
}
```
Example (I): Multiplication Table

```c
#include <stdio.h>

int main()
{
    int n;
    printf("Enter the number: ");
    scanf("%d", &n);

    for(int i = 1 ;i <= n; i++)
    {
        for(int j = 1 ;j <= n; j++)
        {
            printf("%d \t", i*j);
            printf("\n");
        }
    }

    return 0;
}
```
Example (I): Multiplication Table

Input: 10
Example (II): Fibonacci Sequence

• Write a program that for a given n, prints the first n numbers in Fibonacci Sequence

| n  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 | 11 | 12 | 13 | 14 | ...
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>x_n</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>8</td>
<td>13</td>
<td>21</td>
<td>34</td>
<td>55</td>
<td>89</td>
<td>144</td>
<td>233</td>
<td>377</td>
<td>...</td>
</tr>
</tbody>
</table>

The next number is found by adding up the two numbers before it.
Example (II): Fibonacci Sequence

```c
#include<stdio.h>

int main()
{
    int n, first = 1, second = 1, next;
    printf("Enter the number of terms\n");
    scanf("%d",&n);
    printf("First %d terms of Fibonacci series are :-\n",n);
    printf("%d \t %d \t",first, second);
    return 0;
}
```
Example (II): Fibonacci Sequence

```c
#include<stdio.h>

int main()
{
    int n, first = 1, second = 1, next;

    printf("Enter the number of terms\n");
    scanf("%d", &n);
    printf("First %d terms of Fibonacci series are :-\n", n);
    printf("%d 	%d 	%d", first, second);

    int i = 3;
    while (i <= n)
    {
        next = first + second;
        printf("%d 	%d 	%d", next, first, second);
        first = second;
        second = next;
        i = i + 1;
    }
    printf("\n");
    return 0;
}
```
Example (II): Fibonacci Sequence

```c
#include<stdio.h>

int main()
{
    int n, first = 1, second = 1, next;
    printf("Enter the number of terms\n");
    scanf("%d", &n);
    printf("First %d terms of Fibonacci series are :-\n", n);
    printf("%d \t %d \t", first, second);
    
    int i = 3;
    while(i <= n)
    {
        next = first + second;
        printf("%d \t", next);
        i = i + 1;
    }
    printf("\n");
    return 0;
}
```
Example (II): Fibonacci Sequence

```c
#include<stdio.h>

int main()
{
    int n, first = 1, second = 1, next;

    printf("Enter the number of terms\n");
    scanf("%d",&n);
    printf("First %d terms of Fibonacci series are :\n",n);
    printf("%d  %d  %d\n",first, second);

    int i = 3;
    while( i <= n )
    {
        next = first + second;
        printf("%d  %d\n",next);
        first = second;
        second = next;
        i = i + 1;
    }

    printf("\n");
    return 0;
}
```

- Makes variables ready for the next iteration
- Updating while condition
Example (II): Fibonacci Sequence

```
[C++ $gcc fibonacci.cc -o fibonacci.exe
[C++ $./fibonacci.exe
Enter the number of terms
8
First 8 terms of Fibonacci series are :-
1 1 2 3 5 8 13 21
[C++ $]
```
Example (III): GCD

• Write a program that reads two integers from the user and outputs the greatest common divisor.

The greatest common divisor (GCD) of two integers is a largest positive integer (non-zero) that divides the numbers without a remainder.

\[
gcd(21, 9) = 3 \\
gcd(20, 10) = 2
\]
Example (III): GCD

• First attempt:

```c
#include <stdio.h>

int main()
{
    int num1, num2, min;
    printf("Enter two integers: ");
    scanf("%d %d", &num1, &num2);
    ...
    return 0;
}
```
Example (III): GCD

• First attempt:

```c
#include <stdio.h>

int main()
{
    int num1, num2, min;
    printf("Enter two integers: ");
    scanf("%d %d", &num1, &num2);

    /* Find the minimum value of input numbers */
    min = (num1 > num2) ? num2: num1;

    return 0;
}
```
Example (III): GCD

• First attempt:

```c
#include <stdio.h>

int main()
{
    int num1, num2, min;
    printf("Enter two integers: ");
    scanf("%d %d", &num1, &num2);

    /* Find the minimum value of input numbers */
    min = (num1 > num2) ? num2 : num1;

    for(int i = min; i >= 1; --i)
    {
        // Insert code here
    }

    return 0;
}
```
Example (III): GCD

- First attempt:

```c
#include <stdio.h>

int main()
{
    int num1, num2, min;
    printf("Enter two integers: ");
    scanf("%d %d", &num1, &num2);

    /* Find the minimum value of input numbers */
    min = (num1 > num2) ? num2 : num1;

    for(int i = min; i >= 1; --i)
    {
        if( num1%i == 0 && num2%i == 0 )
        {
            printf("GCD of %d and %d is %d\n", num1, num2, i);
        }
    }

    return 0;
}
```
Example (III): GCD

• First failure attempt:

It prints all the numbers both numbers divide
#include <stdio.h>

int main()
{
    int num1, num2, min;
    int found = 0;
    printf("Enter two integers: ");
    scanf("%d %d", &num1, &num2);
    /* Find the minimum value of input numbers */
    min =(num1>num2)? num2: num1;
    for(int i = min; i >= 1 && !found; --i)
    {
        if( num1%i == 0 && num2%i == 0)
        {
            printf("GCD of %d and %d is %d\n", num1, num2, i);
            found = 1;
        }
    }
    return 0;
}
Example (III): GCD

- Correct implementation:

```
[C++] $gcc gcd1.cc -o gcd1.exe
[C++] $./gcd1.exe
Enter two integers: 24 6
GCD of 24 and 6 is 6
[C++] $
```
Omitting expressions

\[ \text{for(}\text{expression1}; \text{expression2}; \text{expression3}) \]

- If second expression is omitted: condition is true – the loop does not terminate (unless stopped in the body)
- Can omit all 3 expressions:
  \[ \text{for (;} ; \text{)} \{\ldots\} \rightarrow \text{while (1)} \{\ldots\} \]
For statement

• Use when we have a counting variable

• Common usages:

  ```java
  for (i =0; i<n; i++) {...}  //Count from 0 to n-1
  for (i =1; i<=n; i++) {...}  //Count from 1 to n
  for (i =n; i>0; i--) {...}   //Count down from n to 1
  for (i =n-1; i>=0; i--) {...} //Count from n-1 to 0
  ```
Altering the Normal Flow of Loops
Altering the Flow

- **When stopping condition is met**
  - **while** and **for**: exit before body executed
  - **do**: exit after body is executed
Altering the Flow

• It is sometimes desirable to skip some statement/s inside loop or terminate the loop immediately without checking the test expression

• Exit in the middle?
  – **break** statement ———> **terminate** the loop
  – **continue** statement ———> **skip** the rest of code in the body of loop
Break
Break

break;

- Causes **immediate exit** from a while, for, do… while or switch statement

- Program execution continues with the **first statement after the structure**

- Common uses of the break statement
  - Escape early from a loop
  - Skip the remainder of a switch statement
Break

- Used to jump out of: while, do, for (an switch)

```c
int sum = 0;
for (;;)
{
    printf("Enter a number\n");
    scanf("%d", n);
    sum += n;
    if (n == 100)
        break;
}
statement 1;
```
Break

break;

• Used to jump out of: while, do, for

```c
int sum = 0;
for (; ;) {
    printf("Enter a number\n");
    scanf("%d", n);
    sum += n;
    if (n == 100)
        break;
}
```
Break

Used to jump out of: while, do, for (an switch)

```c
int sum = 0;
while (1)
{
    printf("Enter a number\n");
    scanf("%d", n);
    sum += n;
    if (n == 100)
    {
        break;
    }
}
```
Break

break;

• Used to jump out of: while, do, for
```c
#include <stdio.h>

int main()
{
    int num1, num2, min;
    int found = 0;

    printf("Enter two integers: ");
    scanf("%d %d", &num1, &num2);

    /* Find the minimum value of input numbers */
    min = (num1 > num2) ? num2 : num1;

    for(int i = min; i >= 1 && !found; --i)
    {
        if( num1%i == 0 && num2%i == 0 )
        {
            printf("GCD of %d and %d is %d\n", num1, num2, i);
            found = 1;
        }
    }
    return 0;
}
```
GCD with break

```c
#include <stdio.h>

int main()
{
    int num1, num2, min;

    printf("Enter two integers: ");
    scanf("%d %d", &num1, &num2);

    /* Find the minimum value of input numbers */
    min = (num1 > num2) ? num2 : num1;

    for(int i = min; i >= 1; --i)
    {
        if( num1 % i == 0 && num2 % i == 0 )
        {
            printf("GCD of %d and %d is %d\n", num1, num2, i);
            break;
        }
    }

    return 0;
}
```
Example

Write a program to find average of maximum of $n$ positive numbers entered by user. But, if the input is negative, display the average (excluding the average of negative input) and end the program.

Input:
Maximum Number: 4
number 1: 4
number 2: -2

Output:
Average: 4
do not need to continue executing loop!
# include <stdio.h>

int main()
{
    float num, average, sum;
    int i, n;
    average = 0;
    sum = 0;
    printf("Maximum number of inputs\n");
    scanf("%d", &n);
    for (i = 1; i <= n; ++i)
    {
        printf("Enter n%d: ", i);
        scanf("%f", &num);
        if (num < 0)
        {
            break; //for loop breaks if num<0
        }
        sum = sum + num;
    }
    average = sum / (i - 1);
    printf("Average=%f\n", average);
    return 0;

Input:
Maximum Number: 4
number 1: 3
number 2: 1
number 3: -1

Output:
Average: 2
Continue
Example

Write a program to find product of 4 integers numbers entered by user. If the user enters 0 skip it, and do not multiply!

Input:
Maximum Number: 4
number 1: 2
number 2: 3
number 3: 0
number 4: 4

Output:
Average: 24
Continue

- Skips the **remaining statements in the body** of a while, for or do…while statement

- Proceeds with the **next iteration** of the loop

- while and do…while
  
  Loop-continuation test is evaluated immediately after the continue statement is executed

- **for**
  
  Increment expression is executed, then the loop-continuation test is evaluated
- Sum of odd numbers between 1 and 100

```java
int sum = 0
for (int i=1; i<100; i++) {
    if (i%2 == 1)
        sum += i;
}
```

```java
int sum = 0
for (int i=1; i<100; i = i+ 2) {
    sum += i;
}
```
• Sum of odd numbers between 1 and 100

```java
int sum = 0
for (int i=1; i<100; i++) {
    if (i%2 == 0) {
        continue;
    }
    sum += i;
}
```
int sum = 0
for (int i=1; i<100; i++)
{
    if (i%2 == 0)
    {
        continue;
    }
    sum += i;
}
int sum = 0
for (int i=1; i<100; i++)
{
    if (i%2 == 0)
        break;
    sum += i;
}

Break
**break and continue Statements**

- **break** immediately exits from a loop
  - Recall its use in the **switch** selection structure

```c
int x;
for(x=1; x<=10; x++)
{
    if(x == 5)
        break;
    printf("%d ", x);
}
printf("Broke out at x == %d\n",x);
1 2 3 4 Broke out at x == 5
```

- **continue** skips remaining statements in the body of a repetition structure

```c
int x, y;
for(x=1; x<=10; x++)
{
    if(x == 5)
    {
        y = x;
        continue;
    }
    printf("%d ", x);
}
printf("Skipped x == %d\n",y);
1 2 3 4 6 7 8 9 10 Skipped x == 5
```
```c
#include <stdio.h>
int main()
{
    int i, num, product;
    product = 1;
    printf("Enter 4 numbers\n");
    for (i = 1; i <= 4; i++)
    {
        printf("Enter num%d: ", i);
        scanf("%d", &num);
        if (num == 0)
        {
            continue;
        }
        product = product* num;
    }
    printf("The product of these number equals:%d\n", product);
    return 0;
```