CS1100 – Introduction to Programming

Lecture 13

Instructor: Shweta Agrawal (shweta.a@cse.iitm.ac.in)

Mini-calculator using While loop

Input operator and two operands, output result till quit.

Mini-calculator using While loop

Input operator and two operands, output result till quit.

```
#include<stdio.h>
int main() {
    int x; int y; char op;
    printf("Input the operator \t"); scanf ("%c", &op);
    while (op != 'q') {
       printf("Input the first integer \t"); scanf ("%d", &x);
       printf("Input the second integer \t"); scanf ("%d", &y);
       switch (op) {
          case '+': printf("x+y = %d\n", x+y); break;
          case '%': printf("x mod y = %d\n", x%y); break;
          case '/': printf("x/y = %.2f\n", (x*0.1)/y); break;
          case 'q': ; break;
          default : printf("invalid operator\n"); break;
       }
       getchar(); printf("Input the operator \t");
       scanf ("%c", &op);
    }
}
```

Mini-calculator: Case Insensitive Quit

```
#include<stdio.h>
int main() {

    // read op.
    while ((op != 'q') || (op != 'Q')) {
        // all the same stuff.
    }
}
```

Mini-calculator: Case Insensitive Quit

```
#include<stdio.h>
int main() {

    // read op.
    while ((op != 'q') || (op != 'Q')) {
        // all the same stuff.
    }
}
```

- Is the condition correct?

Mini-calculator: Case Insensitive Quit

```
#include<stdio.h>
int main() {

    // read op.
    while ((op != 'q') || (op != 'Q')) {
        // all the same stuff.
    }
}
```

- Is the condition correct?
- We want to stop when op is either 'q' or 'Q'.

Mini-calculator: Case Insensitive Quit

```
#include<stdio.h>
int main() {

    // read op.
    while ((op != 'q') || (op != 'Q')) {
        // all the same stuff.
    }
}
```

- Is the condition correct?
- We want to stop when op is either 'q' or 'Q'.
  (op == 'q' || op == 'Q') then quit the loop.

Mini-calculator: Case Insensitive Quit

```
#include<stdio.h>
int main() {

    // read op.
    while ((op != 'q') || (op != 'Q')) {
        // all the same stuff.
    }
}
```

- Is the condition correct?
- We want to stop when op is either 'q' or 'Q'.
  (op == 'q' || op == 'Q') then quit the loop.
- We want to enter the loop when the above condition is false!

Mini-calculator: Case Insensitive Quit

```
#include<stdio.h>
int main() {

    // read op.
    while ((op != 'q') || (op != 'Q')) {
        // all the same stuff.
    }
}
```

- Is the condition correct?
- We want to stop when op is either 'q' or 'Q'.
  (op == 'q' || op == 'Q') then quit the loop.
- We want to enter the loop when the above condition is false!
  !(op == 'q' || op == 'Q')

Mini-calculator: Case Insensitive Quit

```
#include<stdio.h>
int main() {

    // read op.
    while ((op != 'q') || (op != 'Q')) {
        // all the same stuff.
    }
}
```

- Is the condition correct?
- We want to stop when op is either 'q' or 'Q'.
  (op == 'q' || op == 'Q') then quit the loop.
- We want to enter the loop when the above condition is false!
  !(op == 'q' || op == 'Q') same as (op != 'q' && op != 'Q')

Mini-calculator: using a true expression

Let the while loop run forever!

Mini-calculator: using a true expression

Let the while loop run forever!

```
while (1) {
   printf("Input the operator \t"); scanf ("%c", &op);
   printf("Input the first integer \t"); scanf ("%d", &x);
   printf("Input the second integer \t"); scanf ("%d", &y);
   switch (op) {
      case '+': printf("x+y = %d\n", x+y); break;
      // other cases..
   }
   getchar();
   //printf("Input the operator \t");
   //scanf ("%c", &op);
}
```

Mini-calculator: using a true expression

Let the while loop run forever!

```
while (1) {
   printf("Input the operator \t"); scanf ("%c", &op);
   printf("Input the first integer \t"); scanf ("%d", &x);
   printf("Input the second integer \t"); scanf ("%d", &y);
   switch (op) {
      case '+': printf("x+y = %d\n", x+y); break;
      // other cases..
   }
   getchar();
   //printf("Input the operator \t");
   //scanf ("%c", &op);
}
```

- How does the program terminate?
- We can make a check and break out of the loop!

Mini-calculator: using a true expression

Let the while loop run forever and break it when needed.

Mini-calculator: using a true expression

Let the while loop run forever and break it when needed.

```
while (1) {
   printf("Input the operator \t"); scanf ("%c", &op);
   if (op == 'q' || op == 'Q') {
       break;
   }
   printf("Input the first integer \t"); scanf ("%d", &x);
   printf("Input the second integer \t"); scanf ("%d", &y);
   switch (op) {
     case '+': printf("x+y = %d\n", x+y); break;
      // other cases..
   }
   getchar();
}
```

- How does the program terminate?

Mini-calculator: using a true expression

Let the while loop run forever and break it when needed.

```
while (1) {
   printf("Input the operator \t"); scanf ("%c", &op);
   if (op == 'q' || op == 'Q') {
       break;
   }
   printf("Input the first integer \t"); scanf ("%d", &x);
   printf("Input the second integer \t"); scanf ("%d", &y);
   switch (op) {
     case '+': printf("x+y = %d\n", x+y); break;
      // other cases..
   }
   getchar();
}
```

- How does the program terminate?

break takes you out of the current statement (it could be switch / loop).

Stepping back : Stages of Program Design ...

To solve a problem using a computer program :

Stepping back : Stages of Program Design ...

To solve a problem using a computer program :

- Need to develop an idea of the sequence of statements and the flow of the control through the statements to achieve a given task.

Stepping back : Stages of Program Design ...

To solve a problem using a computer program :

- Need to develop an idea of the sequence of statements and the flow of the control through the statements to achieve a given task.
- **Algorithm** : The description of the idea - *a step-by-step procedure to solve the problem*.

Stepping back : Stages of Program Design ...

To solve a problem using a computer program :

- Need to develop an idea of the sequence of statements and the flow of the control through the statements to achieve a given task.
- **Algorithm** : The description of the idea - *a step-by-step procedure to solve the problem*.
- **Pseudo-code :** A language-independent but program-like description of an algorithm.

Stepping back : Stages of Program Design ...

To solve a problem using a computer program :

- Need to develop an idea of the sequence of statements and the flow of the control through the statements to achieve a given task.
- **Algorithm** : The description of the idea - *a step-by-step procedure to solve the problem*.
- **Pseudo-code :** A language-independent but program-like description of an algorithm.
- **Flowchart :** A diagramatic representation of the algorithm is called a *flowchart*.