Slide Type Slide

0.1 ICS 104 - Introduction to Programming in Python and C

0.2 Overview of C - Lab 2

Slide Type Fragment

1 Lab Learning Outcomes

- To develop code using if statements in C.
- To develop loops in C.

Slide Type Slide

2 Examples

Slide Type Fragment

- Example # 1: The following segment needs some revision. Insert braces where they are needed and correct the errors. The corrected code should take five integers and display their sum.

```
count = 0;
while (count <= 5)
count += 1;
printf("Next Number> ");
scanf("%d", &next_num);
next_num += sum;
printf("%d numbers were added; \n", count);
printf("Their sum is %d.\n", sum);
```

In [ ]:

Slide Type Fragment

```
1 /* Corrected Example # 1 */
```

Slide Type Slide

- Example # 2: Trace the execution of the loop that follows for $n = 8$. Show values of odd and sum after the update of the loop counter for each iteration.

```
sum = 0;
for (odd = 1;
   odd < n;
   odd += 2)
  sum = sum + odd;
printf("Sum of the positive odd numbers less than %d is %d.\n", n, sum);
```

Slide Type Fragment

- The output of Example # 2:

Slide Type Slide

- Example # 3: Rewrite the following code using a do-while statement with no decisions in the loop body. In what situations will the rewritten code print an incorrect sum?

```
sum = 0;
for (odd = 1; odd < n;  odd = odd + 2)
  sum = sum + odd;
printf("Sum of the positive odd numbers less than %d is %d.\n", n, sum);
```

Slide Type Fragment

/* Rewritten Code of Example # 3 */

```
#include <stdio.h>

int main()
{
    int sum = 0;
    int odd = 1;
    int n = 8;
    do {
        sum = sum + odd;
        printf("Sum of the positive odd numbers less than or equal to %d is %d.\n", odd, sum);
        odd = odd + 2;
    } while(odd < n);
    return 0;
}
```

input

```
Sum of the positive odd numbers less than or equal to 1 is 1.
Sum of the positive odd numbers less than or equal to 3 is 4.
Sum of the positive odd numbers less than or equal to 5 is 9.
Sum of the positive odd numbers less than or equal to 7 is 16.
```

Slide Type Slide

3 Exercises

Slide Type Fragment

- Exercise # 1: Write a program that determines the day number (1 to 366) in a year for a date that is provided as input data. As an example, January 1,

1994, is day 1. December 31, 1993, is day 365. December 31, 1996, is day 366, since 1996 is a leap year. A year is a leap year if it is divisible by 4, but not by 100 or if it is divisible by 400. Your program should accept the month, day, and year as integers. You may assume that the entered date is correct, i.e. no need for date validation.

- Following are some sample runs:
  - Sample Run 1:

```
Enter date as follows year month day: 1994 1 1
day of the year = 1
```

- Sample Run 2:

```
Enter date as follows year month day: 1996 12 31
day of the year = 366
```

In [ ]:

Slide Type Fragment

```c
/* Exercise # 1 - Source Code */


#include <stdio.h>

int
main ()
{
  int year, month, day, feb, dayNum;

  printf ("Enter date as follows year month day: ");
  scanf ("%d", &year);
  scanf ("%d", &month);
  scanf ("%d", &day);
  if (year % 4 == 0 && year % 100 != 0 || year % 400 == 0)
    feb = 29;
  else
    feb = 28;
  ////////////////////////////////////////////////////////
  if (month == 1)
    dayNum = day;
  else if (month == 2)
    dayNum = 31 + day;

  else if (month == 3)
    dayNum = 31 + feb + day;

  else if (month == 4)
    dayNum = 31 + feb + 31 + day;

  else if (month == 5)
    dayNum = 31 + feb + 31 + 30 + day;

  else if (month == 6)
    dayNum = 31 + feb + 31 + 30 + 31 + day;

  else if (month == 7)
    dayNum = 31 + feb + 31 + 30 + 31 + 30 + day;

  else if (month == 8)
    dayNum = 31 + feb + 31 + 30 + 31 + 30 + 31 +day;

  else if (month == 9)
    dayNum = 31 + feb + 31 + 30 + 31 + 30 + 31 + 31 +day;

  else if (month == 10)
    dayNum = 31 + feb + 31 + 30 + 31 + 30 + 31 + 31 + 30 +day;

  else if (month == 11)
    dayNum = 31 + feb + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31 +day;

  else if (month == 12)
    dayNum = 31 + feb + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 31 + 30 + day;

printf("Day of year: %d", dayNum);

    return 0;
}
```

Slide Type Slide

- Exercise # 2: Write a C program segment that computes $1 + 2 + 3 + \ldots + (n-1) + n$, where n is a positive integer value greater than 1. Assume that you enter an integer, but make sure that you validate that it is a positive integer that is greater than 1. Follow the loop body with an if statement that compares this value to $\frac{n(n+1)}{2}$ and displays a message that indicates whether the values are the same or different. What message do you think will be displayed?
- Following are two sample runs:
  - Sample Run 1:

```
Enter an integer value greater than 1 : 5
sum of numbers from 1 to 5 = 15
```

  - Sample Run 2:

```
Enter an integer value greater than 1 : 0
Wrong input try again
Enter an integer value greater than 1 : -5
Wrong input try again
Enter an integer value greater than 1 : 1
Wrong input try again
Enter an integer value greater than 1 : 20
sum of numbers from 1 to 20 = 210
```

In [ ]:

Slide Type Fragment

```c
/* Exercise # 2 - Source Code */

#include <stdio.h>

int main()
{   int n ,final = 0, x = 0;
    printf("Enter an integer value greater than 1 : ");
    scanf("%d" , &n);

    while (n <= 1){
        printf("Wrong input try again\n");
        printf("Enter an integer value greater than 1 : ");
        scanf("%d" , &n);
    }

    while(x != n){
        x = x + 1;
        final = final + x;
```

```
    }
     if ( final == ((n*(n+1))/2) )
        printf("The values are the same !\n");

    else
        printf("The values are NOT the same!\n");


    printf("sum of numbers from 1 to %d = %d",n,final);

    return 0;
}
```

Slide Type: Slide

- Exercise # 3: Write nests of loops that cause the following output to be displayed:

```
0
0 1
0 1 2
0 1 2 3
0 1 2 3 4
0 1 2 3 4 5
0 1 2 3 4
0 1 2 3
0 1 2
0 1
0
```

In [ ]:

Slide Type: Fragment

```c
/* Exercise # 3 - Source Code */

#include <stdio.h>

int main()
{
    int i ,j;
    for (i = 0; i<6 ; i++)
        {
        for (j = 0; j < i+1 ; j++)
            {
            printf("%d", j);
            }
        printf("\n");
        }
    for (i = 5 ; i > -1 ; i--)
        {
         for (j = 0 ; j < i ; j++)
            {
            printf("%d", j);
            }
        printf("\n");

        }

    return(0);
    }
```