

1310 INLAB 08

Note: Create a new directory by name “lab8”. All the files for inlab 8 should be saved in this directory only. Names of the files should follow the format ‘loop-control-chk1.c’, ‘loop-control-chk2.c’ etc. After you complete a checkpoint, make a copy of that file for the next checkpoint and modify the copied file.

Log into omega and using the 'vi' editor, create a file called 'loop-control-chk1.c'. This program implements the functionalities given below:

1. Reverse an n-digit number, where n can be any positive integer
2. Determine if the number is even or odd
3. Determine if the number is prime
4. Generate a triangle pattern

User is asked to enter a choice, based on which, control is transferred to appropriate case of the switch.

Following is the outline of file ‘loop-control-chk1.c’—

```
/* loop-control-chk1.c - This program requests a positive integer. Functionalities include reversing a number, determining if the number is even or odd, and determining if the number is prime. */
```

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    char ch = 'N'; /* variable to accept the user's choice, N represents the condition not to exit from the program */
```

```
    int num; /* variable to accept a number */
```

```
    while( (ch != 'Y') && (ch != 'y') )
```

```
    {
```

```
        printf("Enter R to reverse the digits of the number \n");
```

```
        printf("Enter E to determine if the number is even or odd \n");
```

```
        printf("Enter P to generate a table of prime numbers\n");
```

```
        printf("Enter T to generate a triangle pattern\n");
```

```
        printf("Enter Y or y to exit the program\n");
```

```
        printf("Enter your choice: ");
```

```
        scanf(" %c", &ch); /*Leave a space before %c first try not leaving a space*/
```

```
        switch(ch)
```

```
        {
```

```
            case 'R':
```

```
            case 'r':
```

```
                printf("Enter the number to be reversed:\n");
```

```
                scanf("%d", &num);
```

```
                break;
```

```
            case 'E':
```

```
            case 'e':
```

```
                printf("Enter the number to check if it is even or odd:\n");
```

```
                scanf("%d", &num);
```

```

        break;
    case 'P':
    case 'p':
        printf("Enter the number to check if it is prime or not:\n");
        scanf("%d", &num);
        break;
    case 'T':
    case 't':
        printf("Enter a number between 0 to 10:\n");
        scanf("%d", &num);
        break;
    case 'Y':
    case 'y':
        printf("Exiting the program.\n");
        break;
    default:
        printf("You have entered a wrong choice. Try again\n");
        break;
} /* switch */
} /* while */
return(0);
} /* main */

```

When you finish running the program without any errors, signal the grader.

Checkpoint 1

In this section you will enter the code for cases 'R' and 'r', which correspond to reversing the digits of a number (any positive integer). Use following 'while' loop to implement this functionality;

```

while(num > 0)
{
    printf("%d", num%10);
    num = num/10;
}

```

Put this code after printf() and scanf() statements of case 'r' of the switch statement and make sure you declare the variable 'num' at the beginning of main(). When program runs without any error, signal the grader.

Checkpoint 2

For the same cases 'R' and 'r', implement the program using a 'do-while' loop. When the code is working, signal the grader.

Checkpoint 3

In this section you will enter code for the cases 'E' and 'e', where you will be checking to see if the number (positive integer) obtained through the keyboard is even or odd. When the code is working, signal the grader.

Hint: A number is even if it is divisible by 2 otherwise it is odd. You can check if a number is divisible by 2 by taking modulo 2 ($\text{num}\%2$) of that number. If the modulus operations return 0, it means that the number is divisible by 2, else it is not. Use an if-else statement for this purpose.

Checkpoint 4

In this checkpoint you will write the code for cases 'P' and 'p' that determines whether a number (positive integer) obtained through the keyboard is prime or not. Use following code to implement this:

```
flag = 0;
for(i=2; i<=num/2; i=i+1)
{
    if(num%i == 0)
    {
        flag = 1;
        break;
    }
}
if(flag == 0)
    printf("Number is prime\n");
else
    printf("Number is not prime\n");
```

You need to declare variables flag and i of type int at the beginning of main(). When the code is compiling without errors and is running, signal the grader.

Checkpoint 5

Convert the for loop in the previous checkpoint(i.e. checkpoint 5) to while loop. For help or hints for conversion ask lab instructor. When the code is compiling without errors and is running, signal the grader.

Checkpoint 6

In this checkpoint you will write the code for cases 'T' and 't' that generates a triangle using * given the height of triangle. Use following code to implement this:

```

for(i=0; i<num; i=i+1)
{
    for(j=i; j<num-1; j=j+1)
        printf(" ");
    for(k=0; k<=i; k=k+1)
        printf("*");
    printf("\n");
}

```

You need to declare variables i, j and k of type int at the beginning of main(). When the code is compiling without errors and is running, signal the grader.

Output:

If num = 5 the following is the output of the above code:

```

*
**
***
****
*****

```

Checkpoint 7

Modify the above code of checkpoint 5 to generate the following triangular pattern:

Output:

If num = 5 the following is the output of the above code:

```

*
***
*****
*****
*****

```

When the code is compiling without errors and is running, signal the grader.

Checkpoint 8

Now add the program header to the above program. Signal the grader when you complete. At this point, your program should be able to compile and execute without any errors, and should also be able to give the required output based on the choices selected from the menu. Signal the grader and display the output of the program.

Checkpoint 9