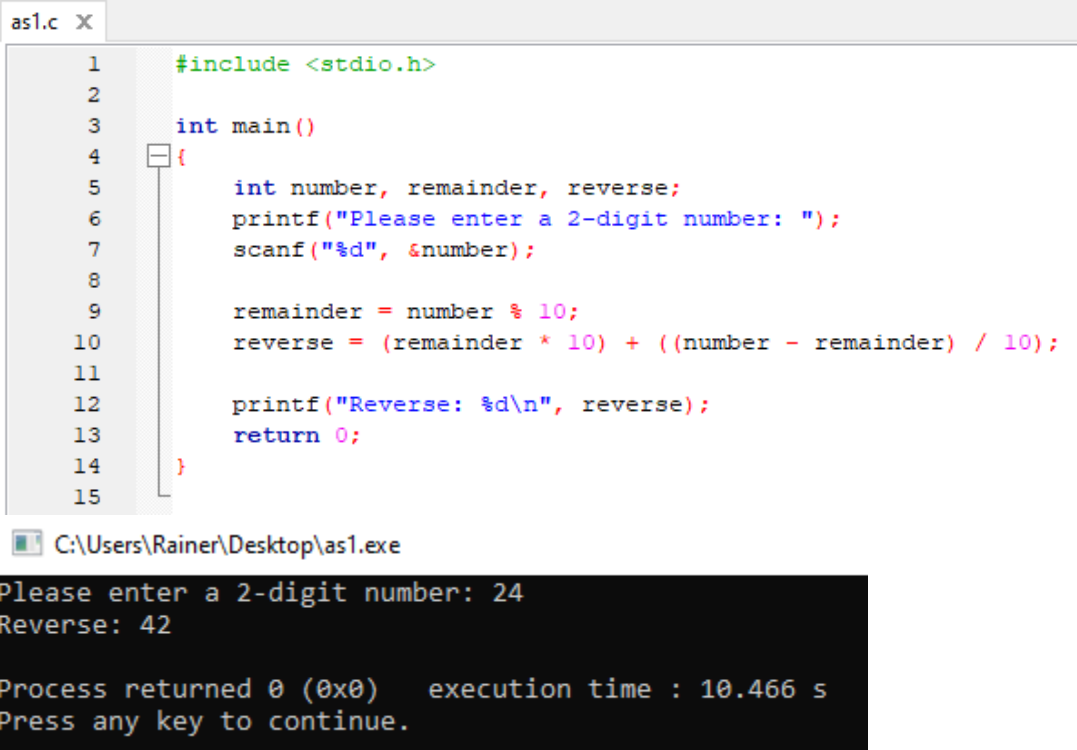


Operators in C

Lecture 1 Assignments

Rainer T. Mayagma
202110417

1. The image shows a C program in a text editor and its execution in a command prompt. The program, named as1.c, is a simple C program that takes a 2-digit number as input and prints its reverse. The code is as follows:

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int number, remainder, reverse;
6      printf("Please enter a 2-digit number: ");
7      scanf("%d", &number);
8
9      remainder = number % 10;
10     reverse = (remainder * 10) + ((number - remainder) / 10);
11
12     printf("Reverse: %d\n", reverse);
13     return 0;
14 }
15
```

The execution output is shown in a black command prompt window. It displays the prompt "Please enter a 2-digit number: 24", followed by the output "Reverse: 42". At the bottom, it shows "Process returned 0 (0x0) execution time : 10.466 s" and "Press any key to continue."

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int number, remainder, reverse;
6      printf("Please enter a 2-digit number: ");
7      scanf("%d", &number);
8
9      remainder = number % 10;
10     reverse = (remainder * 10) + ((number - remainder) / 10);
11
12     printf("Reverse: %d\n", reverse);
13     return 0;
14 }
15
```

C:\Users\Rainer\Desktop\as1.exe

Please enter a 2-digit number: 24
Reverse: 42


Process returned 0 (0x0) execution time : 10.466 s
Press any key to continue.

```

as2.c X
1  #include <stdio.h>
2
3  int main()
4  {
5      int number, remainder1, remainder2, reverse;
6      printf("Please enter a 3-digit number: ");
7      scanf("%d", &number);
8
9      remainder1 = number % 10;
10     number /= 10;
11
12     remainder2 = number % 10;
13     reverse = (remainder1 * 100) + (remainder2 * 10) + ((number - remainder2) / 10);
14
15     printf("Reverse: %d\n", reverse);
16     return 0;
17 }
18

```

2.

 C:\Users\Rainer\Desktop\as2.exe

```

Please enter a 3-digit number: 432
Reverse: 234

```

```

Process returned 0 (0x0)   execution time : 10.935 s
Press any key to continue.

```

```

main.c X
1  #include <stdio.h>
2
3  int main()
4  {
5      int i = 3, j = 4, k = 5;
6      printf("%d", i < j || ++j < k);
7      return 0;
8  }
9

```

3. a.

```

1
Process returned 0 (0x0)   execution time : 0.011 s
Press any key to continue.

```

The output is 1 so the relation is True.

```

main.c X
1      #include <stdio.h>
2
3      int main()
4      {
5          int i = 7, j = 8, k = 9;
6          printf("%d", i - 7 && j++ < k);
7          return 0;
8      }
9

```

b.

```

0
Process returned 0 (0x0)   execution time : 0.016 s
Press any key to continue.

```

The output is 0 so the relation is false.

```

main.c X
1      #include <stdio.h>
2
3      int main()
4      {
5          int i = 7, j = 8, k = 9;
6          printf("%d", (i = j) || (j == k));
7          printf("%d %d %d", i, j, k);
8          return 0;
9      }
10

```

c.

```

18 8 9
Process returned 0 (0x0)   execution time : 0.031 s
Press any key to continue.

```

The output is 1 for the first print function so the relation is True.

The second print function is 8 8 9. It became 8 because in the relation it was stated $i = j$, which $j = 8$.

```

main.c X
1      #include <stdio.h>
2
3      int main()
4      {
5          int i, j, k;
6          i = j = k = 1;
7          printf("%d", ++i || ++j && ++k);
8          printf("%d %d %d", i, j, k);
9          return 0;
10     }
11

```

d.

```

12 1 1
Process returned 0 (0x0)   execution time : 0.016 s
Press any key to continue.

```

The output of the first print function is 1 so the relation is True.

The second print function is became 2 1 1 because of incremenation.