Discovering the reefs of the Sea of Cs

Expressions & Operators

Season 2

CS-101 2021

- We are now in the reefs of the sea of Cs infested by the dangerous BARRACUDAs. **Danger** lurks everywhere!
- Pay close attention to your SCuba Instructor. *Even he* could be prone to attacks!
- Those who wish to C-leep better surface and go back to the shore, else…

A Hand Trace Example

	val	ans
□ main()		
□ {…		
□ int ans, val = 4;		
Code	4	Garbage
□ val = val + 1;	5	
□ val++;	6	
□ ++val;	7	
□ ans = 2 * val++;	8	14
□ ans = ++val / 2;	9	4
□ val--;	8	4
□ --val;	7	4
□ ans = --val * 2;	6	12
□ ans = val-- / 3;	5	2
□ return 0;		
□ }		

Practice...

- // Given:

-
  ```
  int a = 1, b = 2, c = 3, x;
  ```
- // What is the value of x?

-
  ```
  x = ++a * b - c--;
  ```
- // What are the new values of a, b, and c?

```
x = 2*2 - 3
  = 4 - 3
  = 1
```

```
a = 2
b = 2
c = 2
```

More Practice

What is the value of y?
What are the new values of a, b, c, and d?

```
int a = 1, b = 2, c = 3, d = 4, y;
y = ++b / c + a * d++;
```

```
y = 3/3 + 1*4
  = 1 + 4
  = 5
```

a	b	c	d
1	3	3	5

Practice with Assignmen Operators

```
int i = 1, j = 2, k = 3, m = 4;
```

Expression	Value
`i += j + k;`	i=6
`j *= k = m + 5;`	k=9, j=18
`k -= m /= j * 2;`	m=1, k=2

switch

- The switch statement is a multi-way decision that tests whether an expression ***matches one of a number of constant integer values*** and branches accordingly.

```
switch (expression)
{    case const-expr :   statements
     case const-expr :   statements
     default : statements
}
```

  - The type of the expression should be either int or char.

Flowchart of Switch Statement

case a
true
case a action(s)
break
false
case b
true
case b action(s)
break
false
case z
true
case z action(s)
break
false
default action(s)

Multiway Switch Selection example: Simple Calculator

```
int main(){//simple calculator
 int a=50,b=10, R;
 char choice;
 printf(“Enter choice”);
 scanf(“%c”,&choice);

 switch (choice){
  case ‘a’ : R=a+b; printf(“R=%d”,R); break;
  case ‘s’ : R=a-b; printf(“R=%d”,R); break;
  case ‘m’ : R=a*b; printf(“R=%d”,R); break;
  case ‘d’ : R=a/b; printf(“R=%d”,R); break;
  default : printf(“Wrong choice dear\n”); break;
  }
 return 0;
}
```

Multiway Switch Selection example

```
switch (choice){
 case 'A' : // no break, work for both A & a
            // next statement automatically
            // get executed
 case 'a': R=a+b; printf("R=%d",R); break;
 case 'S':
 case 's': R=a-b; printf("R=%d",R); break;
 case 'M':
 case 'm': R=a*b; printf("R=%d",R); break;
 case 'D':
 case 'd': R=a/b; printf("R=%d",R); break;
 default : printf("Wrong choice dear"); break;
 }
```

Range Multiway Switch Selection example

```
int x;
scanf(“%d”,&x);
switch (x){
  case 1 ... 20:// 1 space three dots space 20
      printf(“You entered >=1 and <=20”);
      break;
  case 21 ... 30 :
       printf(“You entered >=21 and <=30”);
      break;
  default   :
      printf(“You entered < 1 and >31”) ;
      break;
  }
```

Syntax = case <low_range> ... <high_range>   :

switch

```
int j;
scanf ("%d", &j);
switch(j) {
        case 0: printf(" zero\n");
        case 1: printf(" one\n");
        case 2: printf(" two\n");
        default: printf(" other\n");
}
```

```
$./a.out
0
zero
one
two
other
$
```

Oops! is this the o/p that we wanted?

- *switch* simply transfers control once to the matching case.

breaking a switch

- break; /*this is a statement which
  can break a switch */
- break exits the switch block.
- break can be used with other control flow structures, but discussion is deferred.

Viewing PKDas's applica...

Use break statements

```
int j;
scanf ("%d", &j);
switch(j) {
    case 0: printf(" zero\n");
            break;
    case 1: printf(" one\n");
            break;
    case 2: printf(" two\n");
    default: printf(" other\n");
}
```

```
$./a.out
0
zero
$./a.out
1
one
$./a.out
2
two
other
$
```

Switch

- default: statements /*optional*/
- The control is transferred to default, if it exists and none of the cases matches the expression value.
- Even if there are multiple statements to be executed in each case there is no need to use { and } (i.e., no need for a compound statement as in if-else).

switch

- You can also use char values.

  -
    ```
    char c ;  c = getchar ( );
            switch (c)
            {
                  case 'a':
                  case 'A': printf("apple");  break;
                  case 'b':
                  case 'B': printf("banana"); break;
            }
    ```

goto

- goto label;

/* label is similar identifier like a variable name */
/* This transfers control to label: */