Cal Poly **CPE 101:** Fundamentals of Computer Science I Alexander Dekhtyar

C Programs: Switch Statement

Switch Statement

Syntax:

```
switch (<expression>) {
   case <label1>: <statement-block1>
   case <label2>: <statement-block2>
    ...
   case <labelN>: <statement-blockN>
   default : <statement-block>
}
```

Notes:

- `default` clause is optional.
- Any `<statement-block>` can be empty.
- `<label1>`,. . . , `<labelN>` are **constant values**.

Semantics:

1. Evaluate `<expression>`.
2. Compare the value of `<expression>` to the value of `<label1>`, `<label2>`,etc, until the comparison is true, or the last `<label>` is reached.
3. If `<expression>` is equal to some `<labelK>`, execute `<statement-blocK>`, `<statement-blockK+1>`,. . . , `<statement-blockN>`, `<statement-block>`.
4. If `<expression>` is not equal to **any** of the `<label1>`,. . . `<labelN>`, and if the `default` keyword is present, execute `<statement-block>`.

Break statement

The behavior of the `switch` statement can be altered with the use of the `break` statement.

Syntax:

```
break;
```

Semantics: End execution of the current { } block, pass control to the first statement immediately following it.

Examples:

```
int main()
{

  int x,y;


  scanf("%d", &x);

  switch (x) {
   case 1: {y = 1; }
   case 2: {y = 2; }
  }

  printf("%d\n",y);

  return 0;
}
```

vs.

```
  int x,y;
  scanf("%d", &x);

  switch (x) {
   case 1: {y = 1; break;}
   case 2: {y = 2; break;}
  }

  printf("%d\n",y);
```