

## Chapter Four: Thinking Logically (Switch Statement)

### Switch, case, break, default

This is a way of making decisions based on many alternatives. Notice carefully the syntax of the program below. The switch statement is made, then in brackets after it has the variable that is examined. Then we must have the alternatives enclosed in { }. Lay out your program the same as this. Each alternative starts with the word “case”, a number and then the alternative followed by colon “:”. At the end of each case is the word “break”, which is to stop the program continuing into the next case statement.

```
switch ( expression )
{ case value1:
  statements
  break;
  case value2:
  statements
  break;
  etc etc etc
  default:
  statements
  break;
}
```

```
public static void main(String[] args)
{
    int num = IBIO.inputInt("enter a number ");
    switch (num)
    { case 1:
      IBIO.output("that number was 1");
      break;
      case 2:
      IBIO.output("that number was 2");
      break;
      default:
      IBIO.output("that number was neither 1 nor 2");
    }
}
```

Pr 4.1 Write a program that will let you enter two numbers. Then it will ask you to enter “1” for add, “2” for multiply, “3” for quit. This will be displayed on the screen like below:

```
Press:  [1]  for addition
        [2]  for multiplication
        [3]  for quit
```

```
public static void main(String[] args)
{
    int a = IBIO.inputInt("enter first number ");
    int b = IBIO.inputInt("enter second number ");
    int num;
    do
    {
        //menu and switch in here
    } while (num != 3)
}
```

If none of these are entered then the program will announce an error and ask again for these possibilities. The program will keep doing this until quit is chosen. Only enter the two numbers once.

- Pr 4.2 Write a program that will add up the sequence  
 $1*7 + 2*2 - 3*5 + 4*7 + 5*2 - 6*5 + 7*7 + 8*2 - 9*5 + 10*7 + \dots 1000$   
Note that there are three cases. First calculate the remainder when divided by 3.  $x\%3$ . If the remainder is 0 then the number gets multiplied by -5, if the remainder is 1 then the number is multiplied by 7 and if the remainder is 2 then the number is multiplied by 2. (669004)

## Labels

The break statement allows the program to exit the current situation (in this case the switch). It is possible to exit more layers. For example we might have a loop which tests alternatives. The program below will continue to loop until the user has guessed the correct number (in this case 17)

```
mainLoop : do
{  int num = IBIO.inputInt("enter a number ");
  switch ( num )
  {  case 17:
      IBIO.output("correct answer");
      break mainLoop;
      default:
      IBIO.output("wrong");
      break;
  }
} while (true);
```

- Pr 4.3 Write a program that allows the user to enter a number less than 1000. The program will search for two numbers that when squared and added together make the number that was input. The program has two loops. One loop going from 1 to *num* and the second loop also goes from 1 to *num* also. In the loop the program will square the numbers and then add them together to see if they are the same as the input number. If they are then use the break statement to break out of both loops. Your output will be the numbers or state that it is impossible.

## Chapter Five: Thinking Abstractly (Decimals)

### double

This is a new data type. Up to now we have only been dealing with whole numbers. Any variable we have used must be declared before we use it.

```
int n; //n is a variable to contain an integer number
```

Now we introduce a new data type – “double”. This can be used to represent a decimal number.

```
public static void main(String[] args)
{
    double a = IBIO.inputDouble("enter first number ");
    double b = IBIO.inputDouble("enter second number ");
    double num = a / b;

    IBIO.output("division gives " + num);
}
```

“output” will print the number. You can see that the accuracy is very high.

### Sequence

The next program will add the numbers together  $1 + 1/3 + 1/9 + 1/27 + 1/81 + \dots$ . It will add 100 of these together. Notice important things about this program. “i” must be declared as an integer as it is used in a “for” statement. The variable “sum” and “term” have both been given descriptive names. In the program we use each term to calculate the next term.

```
public static void main(String[] args)
{
    double term = 1;
    double sum = 0;

    for (int i = 0; i < 100; i++)
    {
        sum = sum + term;
        term = term / 3;
    }

    IBIO.output("total is " + sum);
}
```

Pr 5.1 Write a program that will add up the sequence  
 $1/5 + 1/25 + 1/125 + 1/625 + \dots$  for 100 terms. (0.25)

In the example above we used one term to create the next term. For some sequences this is difficult and it is best to create each new term from scratch as in the next example.

Pr 5.2 Write a program that it adds up the sequence  
 $1/1 + 1/4 + 1/9 + 1/16 + 1/25 + \dots$  for 100 terms (1.6348839001848923)

## Alternating Sequence

If we have an alternating sequence then the problem is more complicated. An alternating sequence is one that the terms alternately add then subtract.

```
double term = 1;
for (int i = 1 ; i < 10 ; i++)
{ IBIO.output(term);
  term = term + 3;
}
```

This creates a sequence of numbers. 1 , 4 , 7 , 10 , 13 ,...

```
double term = 1;
int sign = 1;
for (i = 1 ; i < 10 ; i++)
{ IBIO.output(sign * term);
  term = term + 3;
  sign = sign * -1;
}
```

This creates a sequence of numbers. 1 , -4 , 7 , -10 , 13

## Pi

Pi can be calculated using the formula:

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \dots$$

Pr 5.3 Write a program that will add up the sequence discussed above to 10,000 terms. Output 4 times the answer to get pi. (3.1414926535900345)

$$\frac{\pi}{2} = 1 + \frac{1}{3} + \frac{1 \times 2}{3 \times 5} + \frac{1 \times 2 \times 3}{3 \times 5 \times 7} + \frac{1 \times 2 \times 3 \times 4}{3 \times 5 \times 7 \times 9} + \dots$$

Pr 5.4 The sequence above is a much quicker way of calculating pi Write a program that will add up the sequence above to 100 terms. Output double the answer (3.1415926535897922)