1. Three language constructs are used to manage variables that enable you to check if certain variables exist, remove variables, and check variables’ truth values.
   
i) `isset()`—determines whether a certain variable has already been declared by PHP. It returns a boolean value true if the variable has already been set, and false otherwise, or if the variable is set to the value NULL.

   ii) `unset()`—“undeclares” a previously set variable, and frees any memory that was used by it if no other variable references its value. A call to `isset()` on a variable that has been `unset()` returns false.

   iii) `empty()`—may be used to check if a variable has not been declared or its value is false. This language construct is usually used to check if a form variable has not been sent or does not contain data.

**Example to support the answer:**

```php
<?php
$msg;
if(isset($msg))
    echo "variable exist"."<br>";
else
    echo "var does not exist";
unset($msg);
if(empty($msg))
    echo "hello";
else
    echo "hi";
?>
```

You can have example of your choice.

**Calculator Program:**

```php
<?php
$ch=0; $i=0; $j=0;
do
{
    printf("nPlease enter your choice\n");
    printf("1:Multiply\n");
    printf("2:Divide\n");
    printf("3:Add\n");
    printf("4:Substract\n");
    printf("5.exit\n");
    $ch=fgets(STDIN);
    switch($ch)
    {
```
case 1;
    printf("Please enter two nos\n");
    $i=fgets(STDIN);
    $j=fgets(STDIN);
    echo "Result is";
    echo ($i*$j);
    break;

case 2;
    printf("Please enter two nos\n");
    $i=fgets(STDIN);
    $j=fgets(STDIN);
    echo "Result is";
    echo ($i/$j);
    break;

case 3;
    printf("Please enter two nos\n");
    $i=fgets(STDIN);
    $j=fgets(STDIN);
    echo "Result is";
    echo ($i+$j);
    break;

case 4;
    printf("Please enter two nos\n");
    $i=fgets(STDIN);
    $j=fgets(STDIN);
    echo "Result is";
    echo ($i-$j);
    break;

case 5;
    printf("Exiting\n");
    break;
    default: echo ("Invalid Choice");
}
} while($ch!=5);
?>

2. (a )Equal to compares only the value while identical to compares both value and data type.

```php
<?php
    if(40=="40")
        echo "hi"; // prints hi
    else
        echo "hello";
    if(40=="40")
        echo "hi";
    else
        echo "hello"; // prints hello
?>
```
define("CONSTANT", "Hello world");
echo $CONSTANT;

Output:  Error will be thrown.

(b) Concatenation operator joins multiple strings into a single string.

Program for finding multiplication of first and third digit:

```php
<?php
$n; $a; $b; $c; $d; $mul=1;
$n=fgets(STDIN);
$a=$n/1000;
$e=(int)$a;
$b=$n%1000;
$mul=$mul*$e;
$c=$b%100;
$d=$c/10;
$f=(int)$d;
$mul=$mul*$f;
echo "multiplication of first and third digit is".$mul;
?>
```

Output of the code:

```php
<?php
$a=10; $b=&$a; $b=20; $a=900;
echo $b;
?>
```

will be 900

Output of the code:

```php
$mesg;
$mesg1="hello";
if(isset($mesg))
    echo "I am mesg";
if(isset($mesg1))
    echo "I am mesg1<br>";
if(isset($mesg2))
    echo "I am mesg2";
```

will be I am mesg1

3. (a) Difference between print and echo:
   i) Unlike echo, print can accept only one argument.
   ii) Unlike echo, print returns a value 1 if printing is successful otherwise 0.
Difference between continue and break:
A break statement results in the termination of the statement to which it applies (switch, for, do, or while). A continue statement is used to end the current loop iteration and return control to the beginning of the loop.

Difference between while and do-while:
While is an entry controlled loop in which body of loop get executed only if condition is true while in do-while the body of the do will get executed at least once whether condition is true or false. If the condition is true then the body of do will get executed again. It is an exit controlled loop.

Syntax of while:  
```  while(condition)  
{  Stmt;  Stmt;  Stmt;  }  ```

Syntax of do-while:  
```  do  
{  Stmt;  Stmt;  }  while(condition);  ```

You can take example of your choice to support the answer.

(b)  
```php  
<?php  
$otpay;  
$hour;  
$i=1;  
while($i<=10)  
{  
print 'Enter no of hours worked';  
$hour=fgets(STDIN);  
if($hour>=40)  
{  
$otpay = ($hour - 40)* 12;  
echo 'No of hours worked ='.$hour."<br>";  
echo 'Overtime pay = Rs'.$otpay;  
}  
else  
{  
$otpay = 0;  
echo 'no overtime pay';  
}  
$i++;  
}  
?>  ```

(c) Output of the code will be 12.