INTRO TO C PROGRAMMING: I/O & COMMON PROGRAMMING ERRORS

SECOND PROGRAM

```
#include <stdio.h>
#define pi 3.14159
main()
{
    char charName;
    char charMI='A';
    int intX, intY=10, intZ=10.5;
    float floatNum1=100.25;
    double doubleNum1=10.432432;

    printf("Enter 2 numbers(separated by space): ");
    scanf("%d %d", &intX, &intY);
    printf("intX=%d intY=%d intZ=%d", intX, intY, intZ);

    getchar();
    printf("\n\nEnter your firstname: ");
    scanf("%c", &charName);
    printf("charName=%c charMI=%c", charName, charMI);

    printf("\n\ndoubleNum1=%lf floatNum1=%f
",doubleNum1, floatNum1);
    printf("\npi = %f", pi);
}
/*end of program*/
```

DATA TYPES

- ***VARIABLES***
  - *C has the following data types*
  - *Every variable name must start with a letter; the rest of the name can consist of letters, numbers and underscore characters.*
  - *C recognizes upper and lower case characters as being different.*
  - *You cannot use any of C's keywords like main, while, switch, etc as variable names.*
  - *[Assign: Look for the range of values of each of the data type given below.]*

Type	Use
char	characters
int	integers
float	real numbers
double	large real numbers

- ***CONSTANTS***
  - one can introduce symbolic constants using #define, for example:

```
#define pi 3.14159
```

COMMON C PROGRAMMING ERRORS

- Forgetting to put an ampersand (&) on arguments
  - causes SEGMENTATION FAULT
  - scanf() must have the address of the variable to store input into. This means that often the ampersand address operator is required to compute the addresses. Here's an example:

```
int x;

scanf("%d", x); /*it should be &x*/
```

- Missing operand and using the wrong format for operand
  - C compilers do *not* check that the correct format is used for arguments of a scanf() and printf() call. The most common errors are incompatibilities in the file format used and the variable associated to it.

```
int x;

scanf("%c %d", &x);
```

- Missing closing and terminating characters
  - causes SYNTAX error
  - Omitting a semicolon or a closing brace.
  - Omitting quotation character.

```
#include <stdio.h>

main()
{
   int x;
   float y

   printf("enter an integer: ");
   scanf("%d", &x);
   printf("enter a real number: ";
   scanf("%f, &y);

/*end of program*/
```

- Undeclared variables
  - A variable should be declared before it can be used. The compiler should know the type of the data that can be stored in the variable.

```
main()
{
       int x;

       scanf("%d", &y);
}
```

- Using a forward slash when a backslash is required (for example, substituting "/n" for "\n.")

PRACTICE EXERCISES

- Write a program to display your full name on the monitor.
- Modify the program to display your address and cellphone number on separate lines by adding two additional **printf()** statements.
- Declare an integer variable ***age*** and use this to store your age. Use **scanf()** to get the input from the user. Then, output the age.