INTRODUCTION TO PROGRAMMING IN C - PART 1

Overview

* C is a general-purpose computer programming language
* C is procedural language
* C programs are compiled, not interpreted
Basic C program structure

/* my first program written in C:
   it prints the value of PI and exits */
#include<stdio.h>
#define PI 3.1416
int main()
{
    float pi = PI;
    printf("pi=%f\n", pi);
    return 0;
}

We will use this example to introduce basic concepts
/* my first program written in C;
   it prints the value of PI and exits */

#include<stdio.h>
define PI 3.1416

int main()
{
    float pi =PI;
    printf("pi=%.1f\n", pi);
    return 0;
}
/* my first program written in C; 
it prints the value of PI and exits */
#include<stdio.h>
#define PI 3.1416
int main()
{
    float pi =PI;
    printf("pi=%f\n", pi);
    return 0;
}
/* my first program written in C; 
   it prints the value of PI and exits */
#include<stdio.h>
#define PI 3.1416
int main()
{
    float pi =PI;
    printf("pi=%f\n", pi);
    return 0;
}
/* my first program written in C; 
it prints the value of PI and exits */
#include<stdio.h>
#define PI 3.1416

int main()
{
    float pi =PI;
    printf("pi=%f\n", pi);
    return 0;
}
/* my first program written in C;
   it prints the value of PI and exits */
#include<stdio.h>
define PI 3.1416
int main()
{
    float pi = PI;
    printf("pi=%f\n", pi);
    return 0;
}
/* my first program written in C;  
it prints the value of PI and exits */
#include<stdio.h>
define PI 3.1416
int main()
{
    float pi =PI;
    printf("pi=\%f\n", pi);
    return 0;
}
Basic Input/Output (I/O)

Most basic I/O:

* Output: display characters in monitor (standard output)

* Input: get characters from keyboard (standard input)
Standard output: `printf`

Requires formatted string that specifies:

1) Text to display

2) Format used to display variables

3) Variables to print
Format convention (most common):

<table>
<thead>
<tr>
<th>escape sequence</th>
<th>printf function's interpretation of expression bits</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>%c</code></td>
<td>2’s complement integer printed as an ASCII character</td>
</tr>
<tr>
<td><code>%d</code></td>
<td>2’s complement integer printed as decimal</td>
</tr>
<tr>
<td><code>%e</code></td>
<td>double printed in decimal scientific notation</td>
</tr>
<tr>
<td><code>%f</code></td>
<td>double printed in decimal</td>
</tr>
<tr>
<td><code>%u</code></td>
<td>unsigned integer printed as decimal</td>
</tr>
<tr>
<td><code>%x</code></td>
<td>integer printed as hexadecimal (lower case)</td>
</tr>
<tr>
<td><code>%X</code></td>
<td>integer printed as hexadecimal (upper case)</td>
</tr>
<tr>
<td><code>%%</code></td>
<td>a single percent sign</td>
</tr>
</tbody>
</table>
Examples:

printf("%d is a prime number", 43);
printf("43 + 59 in decimal is %d\n", 43+59);
printf("a+b=%f\n", a+b);
printf("%d+%d=%d\n", a, b, a+b);
Standard input: `scanf`

- Gets data from standard input, typically Keyboard
- Does type conversion from ASCII to specified format

Examples:

```
scanf("%c", &nextchar);
scanf("%f", &radius);
scanf("%d %d", &length, &height);
```