C Programming

C Program

```
/* Volume and surface area of a planet */
#include <stdio.h>
#define PI 3.14159265
main()
{
     float radius, diameter;      // dimensions
     double volume, area;         // size
     char planet[10];

     printf(“Please, enter the name of the planet: “);
     gets(planet);
     printf(“The value of the diameter: “);
     scanf(“%f”, &diameter);
     radius = diameter / 2;
     volume = 4/3 * PI * radius * radius * radius;
     area = 4 * PI * radius * radius;
     printf(“The volume of planet %s is %f km3\n”
     “The %s surface area is %f km2\n”, planet, volume, planet, area);
}
```

C Language

```
/* Volume and surface area of a planet */
#include <stdio.h>
#define PI 3.14159265
main()
{
    float radius, diameter;          // dimensions
    double volume, area;             // size
    char planet[10];

    printf(“Please, enter the name of the planet: “);
    gets(planet);
    printf(“The value of the diameter: “);
    scanf(“%f”, &diameter);
    radius = diameter / 2;
    volume = 4/3 * PI * radius * radius * radius;
    area = 4 * PI * radius * radius;
    printf(“The volume of planet %s is %f km3\n”
    “The %s surface area is %f km2\n”, planet,
        volume, planet, area);
}
```

- Alphabet
  - lowercase and uppercase letters from the English alphabet
  - decimal digits
  - special characters, like
    + - * / ( ) [ ] { } < > / \ = ! ? ‘ “ . , ; : | % & # ^ ~ _ space
  - keywords - text with a special meaning

Reserved Words

auto
break
case
char
const
continue
default
do
double
else
enum
extern
float
for
goto
if
int
long
register
return
short
signed
sizeof
static
struct
switch
typedef
union
unsigned
void
volatile
while

C Language

```
/* Volume and surface area of a planet */
#include <stdio.h>
#define PI 3.14159265
main()
{
    float radius, diameter;          // dimensions
    double volume, area;             // size
    char planet[10];

    printf(“Please, enter the name of the planet: “);
    gets(planet);
    printf(“The value of the diameter: “);
    scanf(“%f”, &diameter);
    radius = diameter / 2;
    volume = 4/3 * PI * radius * radius * radius;
    area = 4 * PI * radius * radius;
    printf(“The volume of planet %s is %f km3\n”
    “The %s surface area is %f km2\n”, planet,
        volume, planet, area);
}
```

- Program structure
  - program title – comment
    - /* */
    - //
  - pre-processor directives
    - #include
    - #define
  - functions

```
main()
{
        block
}
```

C Program

```
/* Volume and surface area of a planet */
#include <stdio.h>
#define PI 3.14159265
main()
{
      float radius, diameter;              // dimensions
      double volume, area;                 // size
      char planet[10];

      printf(“Please, enter the name of the planet: “);
      gets(planet);
      printf(“The value of the diameter: “);
      scanf(“%f”, &diameter);
      radius = diameter / 2;
      volume = 4/3 * PI * radius * radius * radius;
      area = 4 * PI * radius * radius;
      printf(“The volume of planet %s is %f km3\n”
      “The %s surface area is %f km2\n”, planet,
           volume, planet, area);
}
```

- Pre-processor directives
  - #include
  - #define

- for objects to be linked or substituted before the processor executes the program
- .h – header file – for the use of external functions
- define defines symbolic constants and macros

C Program

```
/* Volume and surface area of a planet */
#include <stdio.h>
#define PI 3.14159265
main()
{
    float radius, diameter;         // dimensions
    double volume, area;            // size
    char planet[10];

    printf(“Please, enter the name of the planet: “);
    gets(planet);
    printf(“The value of the diameter: “);
    scanf(“%f”, &diameter);
    radius = diameter / 2;
    volume = 4/3 * PI * radius * radius * radius;
    area = 4 * PI * radius * radius;
    printf(“The volume of planet %s is %f km3\n”
    “The %s surface area is %f km2\n”, planet,
        volume, planet, area);
}
```

- Functions
  - one or more
  - the first / only is called main()
  - all functions have similar structure
    - heading line
    - {
    - *block*
    - }

C Program

```
/* Volume and surface area of a planet */
#include <stdio.h>
#define PI 3.14159265
main()
{
      float radius, diameter;            // dimensions
      double volume, area;               // size
      char planet[10];

      printf(“Please, enter the name of the planet: “);
      gets(planet);
      printf(“The value of the diameter: “);
      scanf(“%f”, &diameter);
      radius = diameter / 2;
      volume = 4/3 * PI * radius * radius * radius;
      area = 4 * PI * radius * radius;
      printf(“The volume of planet %s is %f km3\n”
      “The %s surface area is %f km2\n”, planet,
           volume, planet, area);
}
```

- Functions
  - ▫ main()
    {
    block
    }
  - ▫ block contains two sections:
    - declarations
      - objects definitions
        - ▫ variables
        - ▫ other functions
    - instructions
      - algorithm description
      - C statements

C Program

```
/* Volume and surface area of a planet */
#include <stdio.h>
#define PI 3.14159265
main()
{
    float radius, diameter;         // dimensions
    double volume, area;            // size
    char planet[10];

    printf(“Please, enter the name of the planet: “);
    gets(planet);
    printf(“The value of the diameter: “);
    scanf(“%f”, &diameter);
    radius = diameter / 2;
    volume = 4/3 * PI * radius * radius * radius;
    area = 4 * PI * radius * radius;
    printf(“The volume of planet %s is %f km3\n”
    “The %s surface area is %f km2\n”, planet,
        volume, planet, area);
}
```

- Declarations
  - all objects
  - name / identifier and type
  - before the instructions
  - declaration statements can be used to specify an initial value of the data

data_type name[, name];

Identifiers

- The identifiers are assigned to memory locations and hold the memory needed for variable data values
- Rules for a valid identifier:
  - identifiers are chosen to reflect their use in the program;
  - an identifier consists of a sequence of letters, digits, and underscores;
  - an identifier may not start with a digit;
  - no limitation of the length of an identifier, but the first 31 characters must be unique.

Identifiers

- The identifiers are assigned to memory locations and hold the memory needed for variable data values
- Rules for a valid identifier:
    - identifiers are chosen to reflect their use in the program;
    - an identifier consists of a sequence of letters, digits, and underscores;
    - an identifier may not start with a digit;
    - no limitation of the length of an identifier, but the first 31 characters must be unique.