Tutorial Link https://codequotient.com/tutorials/Void Pointers/5a008a80cbb2fe34b7774faf

TUTORIAL

Void Pointers

Chapter

1. Void Pointers

Topics

1.2 Useful points about void pointers

A void pointer is a pointer that has no associated data type with it. A void pointer can hold address of any type and can be type casted to any type.

```
int var1 = 10;
char var2 = 'x';
void *p = &var1;    // void pointer holds address of int var1
p = &var2;       // void pointer holds address of char var2
```

malloc() and calloc() return void * type and this allows these functions to be used to allocate memory of any data type (just because of void *)

```
int main()
{
        int *x = (int*) malloc(sizeof(int) * n);      //
typecasting void pointer to int pointer.
}
```

Useful points about void pointers

Some useful points about void pointers are:

1) void pointers cannot be dereferenced. For example the following program has error: -.

```c
#include<stdio.h>

int main()
{
  int a = 10;
  void *ptr = &a;
  printf("%d", *ptr);    // Error, as void pointer cannot be dereferenced
  return 0;
}
```

The following program compiles and runs fine.

```c
#include<stdio.h>

int main()
{
  int a = 10;
  void *ptr = &a;
  printf("%d", *(int *)ptr);        // void pointer type casted to int and then used is fine.
  return 0;
}
```

Output:

```
10
```

2) The C standard doesn't allow pointer arithmetic with void pointers. However, in GNU C it is allowed by considering the size of void is 1. For example the following program compiles with warning but runs in gcc. (Although this behavior is also dependent on gcc implementation i.e. it may not run on some gcc versions also).

```
#include<stdio.h>

int main()
{
  int a[2] = {1, 2};
  void *ptr = &a;
  ptr = ptr + sizeof(int);   // move to second index of array
  printf("%d", *(int *)ptr);       // dereference after casting in
integer pointer.
  return 0;
}
```

Output:

```
2
```

Note that the above program may not work in other compilers. In above program, if you use normal increment with void pointers then results will be unpredictable as below: -

```
#include<stdio.h>

int main()
{
  int a[2] = {1, 2};
  void *ptr = &a;
  ptr = ptr + 1;           // move to a garbage address
  printf("%d", *(int *)ptr);       // dereference after casting in
integer pointer.
```

```
  return 0;
}
```

Output will be a garbage value:

```
278462
```

codequotient.com

Tutorial by codequotient.com | All rights reserved, CodeQuotient 2023