```c
// Storage Classes in C

// auto , default , until scope of function

// extern

// Space allocated , not initialized

// static

// initialized to 0 , value remains till program terminates

// register , same as auto , computer tries, tries

//   to store in free register

//   address of register variable not availabe

// A C program to demonstrate different storage
// classes
#include <stdio.h>

#include "extern.c"

// declaring and initializing an extern variable
extern int x;

// declaring and initialing a global variable z
// simply int z; would have initialized z with
// the default value of a global variable which is 0
int z = 10;


int main()
{
    // declaring an auto variable (simply
    // writing "int a=32;" works as well)
    auto int a = 32;

    // declaring a register variable
    register char b = 'G';

    // b = "G" difference ?

    // telling the compiler that the variable
    // z is an extern variable and has been
    // defined elsewhere (above the main
    // function)
    extern int z;

    printf("Hello World!\n");

    // printing the auto variable 'a'
    printf("\nThis is the value of the auto "
           " integer 'a': %d\n",a);

    // printing the extern variables 'x'
    // and 'z'
    printf("\nThese are the values of the"
           " extern integers 'x' and 'z'"
           " respectively: %d and %d\n", x, z);

    // printing the register variable 'b'
    printf("\nThis is the value of the "
           "register character 'b': %c\n",b);

    // value of extern variable x modified
    x = 2;
    // value of extern variable z modified
    z = 5;

    // printing the modified values of
    // extern variables 'x' and 'z'
    printf("\nThese are the modified values "
           "of the extern integers 'x' and "
           "'z' respectively: %d and %d\n",x,z);

    // using a static variable 'y'
    printf("\n'y' is a static variable and its "
           "value is NOT initialized to 5 after"
           " the first iteration! See for"
           " yourself :)\n");

    while (x > 0)
    {
        static int y = 5;
        y++;

        // printing value of y at each iteration
        printf("The value of y is %d\n",y);
        x--;
    }

    return 0;
}
```