<https://www.onlinegdb.com/online_c++_compiler>

Just testing how pixel buffering works

| /**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****\*\*\**   *Online C++ Compiler.*  *Code, Compile, Run and Debug C++ program online.* Write your code in this editor and press "Run" button to compile and execute it.  **\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****\*\*\**\*/  #include <iostream>  using namespace std;  #define BYTE\_TO\_BINARY\_PATTERN "%c%c%c%c%c%c%c%c" #define BYTE\_TO\_BINARY(byte) \  (byte & 0x80 ? '1' : '0'), \  (byte & 0x40 ? '1' : '0'), \  (byte & 0x20 ? '1' : '0'), \  (byte & 0x10 ? '1' : '0'), \  (byte & 0x08 ? '1' : '0'), \  (byte & 0x04 ? '1' : '0'), \  (byte & 0x02 ? '1' : '0'), \  (byte & 0x01 ? '1' : '0')   int main() {    unsigned char pixelbuffer = 0;  pixelbuffer = pixelbuffer >> 1 | 0x80;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN"\n", BYTE\_TO\_BINARY(pixelbuffer));  pixelbuffer = pixelbuffer >> 1;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN"\n", BYTE\_TO\_BINARY(pixelbuffer));  pixelbuffer = pixelbuffer >> 1 | 0x80;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN"\n", BYTE\_TO\_BINARY(pixelbuffer));  pixelbuffer = pixelbuffer >> 1 | 0x80;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN"\n", BYTE\_TO\_BINARY(pixelbuffer));  pixelbuffer = pixelbuffer >> 1;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN"\n", BYTE\_TO\_BINARY(pixelbuffer));  pixelbuffer = pixelbuffer >> 1 | 0x80;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN"\n", BYTE\_TO\_BINARY(pixelbuffer));  pixelbuffer = pixelbuffer >> 1 | 0x80;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN"\n", BYTE\_TO\_BINARY(pixelbuffer));  pixelbuffer = pixelbuffer >> 1 | 0x80;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN"\n", BYTE\_TO\_BINARY(pixelbuffer));  pixelbuffer = 0;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN"\n", BYTE\_TO\_BINARY(pixelbuffer));    return 0; } |
| --- |

| /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   Online C++ Compiler.  Code, Compile, Run and Debug C++ program online. Write your code in this editor and press "Run" button to compile and execute it.  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ #include <stdio.h>  void print\_binary(unsigned int number) {  if (number >> 1) {  print\_binary(number >> 1);  }  putc((number & 1) ? '1' : '0', stdout); }  //#include <iostream>  //using namespace std;  #define BYTE\_TO\_BINARY\_PATTERN "%c%c%c%c %c%c%c%c %c%c%c%c %c%c%c%c %c%c%c%c %c%c%c%c %c%c%c%c %c%c%c%c" #define BYTE\_TO\_BINARY(byte) \  (byte & 0x80000000 ? '1' : '0'), \  (byte & 0x40000000 ? '1' : '0'), \  (byte & 0x20000000 ? '1' : '0'), \  (byte & 0x10000000 ? '1' : '0'), \  (byte & 0x08000000 ? '1' : '0'), \  (byte & 0x04000000 ? '1' : '0'), \  (byte & 0x02000000 ? '1' : '0'), \  (byte & 0x01000000 ? '1' : '0'), \  (byte & 0x00800000 ? '1' : '0'), \  (byte & 0x00400000 ? '1' : '0'), \  (byte & 0x00200000 ? '1' : '0'), \  (byte & 0x00100000 ? '1' : '0'), \  (byte & 0x00080000 ? '1' : '0'), \  (byte & 0x00040000 ? '1' : '0'), \  (byte & 0x00020000 ? '1' : '0'), \  (byte & 0x00010000 ? '1' : '0'), \  (byte & 0x00008000 ? '1' : '0'), \  (byte & 0x00004000 ? '1' : '0'), \  (byte & 0x00002000 ? '1' : '0'), \  (byte & 0x00001000 ? '1' : '0'), \  (byte & 0x00000800 ? '1' : '0'), \  (byte & 0x00000400 ? '1' : '0'), \  (byte & 0x00000200 ? '1' : '0'), \  (byte & 0x00000100 ? '1' : '0'), \  (byte & 0x00000080 ? '1' : '0'), \  (byte & 0x00000040 ? '1' : '0'), \  (byte & 0x00000020 ? '1' : '0'), \  (byte & 0x00000010 ? '1' : '0'), \  (byte & 0x00000008 ? '1' : '0'), \  (byte & 0x00000004 ? '1' : '0'), \  (byte & 0x00000002 ? '1' : '0'), \  (byte & 0x00000001 ? '1' : '0')   int main() {    unsigned int pixelbuffer = 0;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN "\n", BYTE\_TO\_BINARY(pixelbuffer));  pixelbuffer = pixelbuffer >> 1 | 0x80;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN "\n", BYTE\_TO\_BINARY(pixelbuffer));  pixelbuffer = pixelbuffer >> 1;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN "\n", BYTE\_TO\_BINARY(pixelbuffer));  pixelbuffer = pixelbuffer >> 1 | 0x80;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN "\n", BYTE\_TO\_BINARY(pixelbuffer));  pixelbuffer = pixelbuffer >> 1 | 0x80;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN "\n", BYTE\_TO\_BINARY(pixelbuffer));  pixelbuffer = pixelbuffer >> 1;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN "\n", BYTE\_TO\_BINARY(pixelbuffer));  pixelbuffer = pixelbuffer >> 1 | 0x80;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN "\n", BYTE\_TO\_BINARY(pixelbuffer));  pixelbuffer = pixelbuffer >> 1 | 0x80;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN "\n", BYTE\_TO\_BINARY(pixelbuffer));  pixelbuffer = pixelbuffer >> 1 | 0x80;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN "\n", BYTE\_TO\_BINARY(pixelbuffer));  pixelbuffer = 0;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN "\n", BYTE\_TO\_BINARY(pixelbuffer));    pixelbuffer = 0xFFFFFF;  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN "\n", BYTE\_TO\_BINARY(pixelbuffer));  print\_binary(0xFFFF);  printf("\n");    unsigned int a = 0b100100;  printf("%d\n",a);  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN "\n", BYTE\_TO\_BINARY(a));  a<<=1;  printf("%d\n",a);  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN "\n", BYTE\_TO\_BINARY(a));  a<<=1;  printf("%d\n",a);  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN "\n", BYTE\_TO\_BINARY(a));  a<<=1;  printf("%d\n",a);  printf("Leading text " BYTE\_TO\_BINARY\_PATTERN "\n", BYTE\_TO\_BINARY(a));  return 0; } |
| --- |