# PROGRAM C

|  |
| --- |
| /\* File hello.c \*/ |
| void main()  {  printf("hello\n ");  } |

|  |
| --- |
| /\* File : hello1.c \*/ |
| /\* menuliskan hello ke layar \*/  /\* pola ini merupakan standard yang dipakai di kelas \*/ int  main ()  {  /\* KAMUS \*/  /\* ALGORITMA \*/ printf ("hello\n"); return 0;  } |

# ASSIGNMENT, INPUT & OUTPUT

|  |
| --- |
| /\* File : ASIGN.C \*/ |
| /\* Assignment nilai integer dan print \*/ int  main ()  {/\* Kamus \*/ int i;  /\* Program \*/ printf ("hello\n"); i = 5;  printf ("Ini nilai i : %d \n", i); return 0;  } |

|  |
| --- |
| /\* File asgdll.c \*/ |
| int main()  { /\* Kamus \*/ float f;  long double fll;  /\* Algoritma \*/ f= 20.0f; fll=10.0L;  return 0;  } |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| /\* File : ASIGNi.C \*/ | | | | |
| /\* | Assignement | dan | print | \*/ |

|  |
| --- |
| #include <limits.h> int  main ()  {  /\* Kamus \*/ int i;  long int ii;  /\* Program \*/ printf ("hello\n"); i = 1234;  ii = 123456;  printf ("Ini nilai i=1234 = : %d \n", i); printf ("Ini nilai ii=123456 : : %10d \n", ii);  /\* print nilai batas integer \*/  printf (“Min dan Max integer : %d, %d \n”, INT\_MIN, INT\_MAX); printf (“Max long integer : %ld, %ld \n”, LONG\_MAX);  return 0;  } |

|  |
| --- |
| /\* File : asign2.c \*/ |
| /\* Deskripsi : \*/  /\* Program ini berisi contoh sederhana untuk mendefinisikan \*/  /\* variabel-variabel bilangan bulat (short int, int, long int),  \*/  /\* karakter, bilangan riil, \*/  /\*  \*/ int  main ()  {  /\* KAMUS \*/ short ks = 1; int ki = 1;  long kl = 10000;  char c = 65; /\* inisialisasi karakter dengan integer  \*/  char c1 = 'Z'; /\* inisialisasi karakter dengan karakter \*/ float x = 1.55;  /\* Algoritma \*/  /\* penulisan karakter sebagai karakter \*/ printf ("Karakter = %c\n", c);  printf ("Karakter = %c\n", c1);  /\* penulisan karakter sebagai integer \*/ printf ("Karakter = %d\n", c);  printf ("Karakter = %d\n", c1);  printf ("Bilangan integer (short) = %d\n", ks); printf ("\t\t(int) = %d\n", ki);  printf ("\t\t(long) = %ld\n", kl); /\* perhatikan format %ld \*/ printf ("Bilangan Real = %f8.3\n", x);  return 0;  } |

# OPERATOR & EKSPRESI

|  |
| --- |
| /\* File : incr.c \*/ |
| /\* Effek dari operator ++ \*/ int  main ()  {  /\* Kamus \*/ int i, j;  /\* Program \*/ i = 3;  j = i++;  printf ("Nilai i : %d\nNilai j : %d\n", ++i, j); return 0;  } |

|  |
| --- |
| /\* File : oper1.c \*/ |
| /\* pemakaian beberapa operator terhadap bit \*/ int  main ()  {  /\* KAMUS \*/  int n = 10; /\* 1010 \*/  int x = 1; /\* 1 \*/  int y = 2; /\* 10 \*/  /\* ALGORITMA \*/  printf ("n = %d \n", n);  printf ("x = %d \n", x);  printf ("y = %d \n", y);  printf ("n & 8 = %d \n", n & 8); /\* 1010 AND 1000 \*/ printf ("x & ~ 8 = %d \n", x & ~8); /\* 1 AND 0111 \*/ printf ("y << 2 = %d \n", y << 2); /\* 10 ==> 1000 = 8 \*/ printf ("y >> 3 = %d \n", y >>3);; /\* 10 ==> 0000 = 0 \*/  return 0;  } |

|  |
| --- |
| /\* File : oprator.c \*/ |
| /\* Contoh pengoperasian variabel bertype dasar \*/ int main ()  #include "boolean.h"  {/\* Kamus \*/  boolean Bool1, Bool2, TF ; int i,j, hsl ;  float x,y,res;  /\* algoritma \*/  printf ("Utk program ini, baca teksnya dan tambahkan output"); Bool1 = true; Bool2 = false;  TF = Bool1 && Bool2 ; /\* Boolean AND \*/ TF = Bool1 || Bool2 ; /\* Boolean OR \*/ TF = ! Bool1 ; /\* NOT \*/  TF = Bool1 ^Bool2; /\* XOR \*/  /\* operasi numerik \*/ i = 5; j = 2 ;  hsl = i+j; hsl = i - j; hsl = i / j; hsl = i \* j; hsl = i /j ; /\* pembagian bulat \*/  hsl = i%j ; /\* sisa. modulo \*/  /\* operasi numerik \*/ x = 5.0 ; y = 2.0 ;  res = x + y; res = x - y; res = x / y; res = x \* y;  /\* operasi relasional numerik \*/ TF = (i==j); TF = (i!=j);  TF = (i < j); TF = (i > j); TF = (i <= j); TF = (i >= j);  /\* operasi relasional numerik \*/ TF = (x != y);  TF = (x < y); TF = (x > y); TF = (x <= y); TF = (x >= y);  return 0;  } |

|  |
| --- |
| /\* File : oper2.c \*/ |
| /\* pemakaian beberapa operator terhadap RELATIONAL DAN bit \*/ int main ()  {/\* KAMUS \*/ char i, j;  /\* ALGORITMA \*/  i = 3; /\* 00000011 dalam biner \*/  j = 4; /\* 00000100 dalam biner \*/  printf ("i = %d \n", i);  printf ("j = %d \n", j);  printf (" i && j = %d \n", i && j);/\* 1:op lojik : true and true=> true \*/  printf (" i & j = %d \n", i & j); /\* 0: 00000000 dalam biner \*/ printf (" i|| j = %d \n", i || j); /\* 1 : OR LOJIK, True or true  => true \*/  printf (" i| j = %d \n", i | j); /\* 7: 00000111 biner \*/ printf (" i^j = %d \n", i ^ j); /\* 7: 00000111 biner \*/ printf (" ~i = %d \n", ~i); /\* -4: 11111100 biner \*/ return 0;  } |

Perhatikanlah operator boolean dibandingkan operator bit jika dipakai dalam ekspresi kondisional:

|  |
| --- |
| /\* KAMUS \*/ char i, j; i=3; j=4;  /\* Algoritma \*/  if (i&&j) {…}; /\* true = true and true \*/ if (i&j) {…}; /\* false = 00000000 \*/  if (i||j) {…}; /\* true = true and true \*/ if (i|j) {…}; /\* true = 00000111 \*/  if (i^j) {…}; /\* true = 00000111 \*/ if (~i) {…}; /\* true = 11111000 \*/ |

|  |  |  |
| --- | --- | --- |
| /\* File : exp.c \*/ | | |
| /\* pemakaian operator kondisional \*/ |  |  |
| int |  |  |
| main () |  |  |
| { |  |  |
| /\* KAMUS \*/ |  |  |
| int x = 1; |  |  |
| int y = 2; |  |  |
| /\* ALGORITMA \*/ |  |  |
| printf ("x = %d \n", x); |  |  |
| printf ("y = %d \n", y); |  |  |
| printf ("hasil ekspresi = (x<y)?x:y | = | %d \n", (x < y) ? x : y); |
| return 0; |  |  |
| } |  |  |

|  |
| --- |
| /\* File : exp1.c \*/ |
| /\* pembagian integer, casting \*/ int main ()  {/\* KAMUS \*/ int x = 1; int y = 2; float fx; float fy;  /\* ALGORITMA \*/  printf ("x/y (format integer) = %d \n", x/y); printf ("x/y (format float) = %f \n", x/y);  /\* supaya hasilnya tidak nol \*/ fx=x;  fy=y;  printf ("x/y (format integer) = %d \n", fx/fy); printf ("x/y (format float) = %f \n", fx/fy);  /\* casting \*/  printf ("float(x)/float(y) (format integer) = %d \n", (float)x/(float)y);  printf ("float(x)/float(y) (format float) = %f \n", (float)x/(float)y);  x = 10;  y = 3;  printf ("x/y (format integer) = %d \n", x/y); printf ("x/y (format float) = %f \n", x/y); return 0;  } |

|  |
| --- |
| /\* File : oper3.c \*/ |
| /\* Operator terner \*/  /\* Ekspresi ditulis sebagai makro \*/ #define max(a,b) ((a>b) ? a: b)  int main ()  {  /\* KAMUS \*/  int i = 0; /\* perhatikan int i,j=0 bukan seperti ini \*/ int j = 0;  char c = 8; char d = 10;  char e = max (c, d); int k = max (i, j);  /\* ALGORITMA \*/  printf ("Nilai e = %d \n", e); printf ("Nilai k = %d \n", k); i = 2;  j = 3;  k = max (i++, j++);  printf ("Nilai k = %d \n", k); return 0;  } |

# PEMBACAAN NILAI

|  |
| --- |
| /\* File : BACA.C \*/ |
| /\* contoh membaca integer \*/  /\* kemudian menuliskan nilai yang dibaca \*/ int main ()  {/\* Kamus \*/ int a;  /\* Program \*/  printf ("Contoh membaca dan menulis, ketik nilai integer: "); scanf ("%d", &a);  printf ("Nilai yang dibaca : %d \n", a); return 0;  } |

|  |
| --- |
| /\* file : bacakar.c \*/ |
| int main()  {/\* Kamus \*/ char cc;  /\* Algoritma \*/ printf ("hello\n");  printf("baca 1 kar : ");scanf ("%c ", cc); printf ("%c", cc);  printf ("bye \n"); return 0;  } |

|  |
| --- |
| /\* File : BACASTR.C \*/ |
| /\* deklarasi dan alokasi string, kemudian mengisinya dengan membaca  \*/ int  main () |

|  |
| --- |
| {  /\* Kamus \*/ char \*str; char \*str1;  /\* Program \*/  printf ("\nBaca string, maks 20 karakter: "); str = (char \*) malloc (20 \* sizeof (char));  printf("masukkan sebuah string, max 20 kar: "); scanf("%s",str); printf ("String yang dibaca : %s\n", str);  str1 = (char \*) malloc (20 \* sizeof (char)); strcpy (str1, str);  printf ("String yang disalin : %s\n", str1); return 0;  } |

|  |
| --- |
| /\* File : asgSTR.C \*/ |
| /\* deklarasi dan alokasi string, kemudian mengisinya dengan membaca  \*/ int  main ()  {  /\* Kamus \*/ char \*str; char \*str1;  /\* Program \*/  printf ("\nBaca string, maks 20 karakter: "); str = (char \*) malloc (20 \* sizeof (char)); strcpy(str,"Ini string..");  printf ("String yang diisikan : %s\n", str); str1 = (char \*) malloc (20 \* sizeof (char)); strcpy (str1, str);  printf ("String yang disalin : %s\n", str1); return 0;  } |