

C Fundamentals and Console I/O

CSC230: C and Software Tools
N.C. State Department of Computer Science



CSC230 - C and Software Tools © NC State University Computer Science Faculty

Outline

- C Coding Style
 - **indent**
- Executing Java and C Programs
- Platform Independence?
- Just-in-Time Compilation
- C Compilation Steps
 - **gcc**
- C99 and C89
- Console I/O
- Streams
- Character I/O
 - **printf**



CSC230 - C and Software Tools © NC State University Computer Science Faculty

C Coding Style (Conventions)

- Universal agreement
 1. clarity and consistency important
 2. indentation, white space, and comments helpful
 3. consistent naming conventions helpful
- See the Style Guidelines for CSC230

Tools (intelligent editors, `indent`, etc.) will take care of much formatting for you

Does it Matter?

- Entries from the
International Obfuscated C Code (IOCC)
Contest...

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define w "Hk-HdA=Jk|Jk-LsYL[MwMxNksNss:"
#define r "Ht@H|@HdHJtJHdYHY:HTFHF=JDBIL"
#define S "IT@I\@=HdHHTGH|KILJJDJDW-WIKID\
"K=HdQHTPH|TIDRJDRJQJ:JC7JK7=JDRJLRI|UITU:8T"
#define _ (i,j)L[i]=2*T[j],0[i]=0[j]-R[j],T[i]=2*
R[j]-5*T[j]+4*0[j]-L[j],R[i]=3*T[j]-R[j]-3*0[j]+L[j],
#define t "IS?I\@=HdGHTGIDJILJDIITHTJTFJDF:8J"

#define y yy(4),yy(5), yy(6),yy(7)
#define yy(
1)R[i]=T[i],T[i]
=0[i],0[i]=L[i]
#define Y _ (0
],4] _ (1],5] _ (2
],6] _ (3],7] _ (4
],8] _ (5],9] _ (6
],10] _ (7],11] _ (8
],12] _ (9],13] _ (10
],14] _ (11],15] _ (12
],16] _ (13],17] _ (14
],18] _ (15],19] _ (16
],20] _ (17],21] _ (18
],22] _ (19],23] _ (20
],24] _ (21],25] _ (22
],26] _ (23],27] _ (24
],28] _ (25],29] _ (26
],30] _ (27],31] _ (28
],32] _ (29],33] _ (30
],34] _ (31],35] _ (32
],36] _ (33],37] _ (34
],38] _ (35],39] _ (36
],40] _ (37],41] _ (38
],42] _ (39],43] _ (40
],44] _ (41],45] _ (42
],46] _ (43],47] _ (44
],48] _ (45],49] _ (46
],50] _ (47],51] _ (48
],52] _ (49],53] _ (50
],54] _ (51],55] _ (52
],56] _ (53],57] _ (54
],58] _ (55],59] _ (56
],60] _ (57],61] _ (58
],62] _ (59],63] _ (60
],64] _ (61],65] _ (62
],66] _ (63],67] _ (64
],68] _ (65],69] _ (66
],70] _ (67],71] _ (68
],72] _ (69],73] _ (70
],74] _ (71],75] _ (72
],76] _ (73],77] _ (74
],78] _ (75],79] _ (76
],80] _ (77],81] _ (78
],82] _ (79],83] _ (80
],84] _ (81],85] _ (82
],86] _ (83],87] _ (84
],88] _ (85],89] _ (86
],90] _ (87],91] _ (88
],92] _ (89],93] _ (90
],94] _ (91],95] _ (92
],96] _ (93],97] _ (94
],98] _ (95],99] _ (96
],100] _ (97],101] _ (98
],102] _ (99],103] _ (100
],104] _ (101],105] _ (102
],106] _ (103],107] _ (104
],108] _ (105],109] _ (106
],110] _ (107],111] _ (108
],112] _ (109],113] _ (110
],114] _ (111],115] _ (112
],116] _ (113],117] _ (114
],118] _ (115],119] _ (116
],120] _ (117],121] _ (118
],122] _ (119],123] _ (120
],124] _ (121],125] _ (122
],126] _ (123],127] _ (124
],128] _ (125],129] _ (126
],130] _ (127],131] _ (128
],132] _ (129],133] _ (130
],134] _ (131],135] _ (132
],136] _ (133],137] _ (134
],138] _ (135],139] _ (136
],140] _ (137],141] _ (138
],142] _ (139],143] _ (140
],144] _ (141],145] _ (142
],146] _ (143],147] _ (144
],148] _ (145],149] _ (146
],150] _ (147],151] _ (148
],152] _ (149],153] _ (150
],154] _ (151],155] _ (152
],156] _ (153],157] _ (154
],158] _ (155],159] _ (156
],160] _ (157],161] _ (158
],162] _ (159],163] _ (160
],164] _ (161],165] _ (162
],166] _ (163],167] _ (164
],168] _ (165],169] _ (166
],170] _ (167],171] _ (168
],172] _ (169],173] _ (170
],174] _ (171],175] _ (172
],176] _ (173],177] _ (174
],178] _ (175],179] _ (176
],180] _ (177],181] _ (178
],182] _ (179],183] _ (180
],184] _ (181],185] _ (182
],186] _ (183],187] _ (184
],188] _ (185],189] _ (186
],190] _ (187],191] _ (188
],192] _ (189],193] _ (190
],194] _ (191],195] _ (192
],196] _ (193],197] _ (194
],198] _ (195],199] _ (196
],200] _ (197],201] _ (198
],202] _ (199],203] _ (200
],204] _ (201],205] _ (202
],206] _ (203],207] _ (204
],208] _ (205],209] _ (206
],210] _ (207],211] _ (208
],212] _ (209],213] _ (210
],214] _ (211],215] _ (212
],216] _ (213],217] _ (214
],218] _ (215],219] _ (216
],220] _ (217],221] _ (218
],222] _ (219],223] _ (220
],224] _ (221],225] _ (222
],226] _ (223],227] _ (224
],228] _ (225],229] _ (226
],230] _ (227],231] _ (228
],232] _ (229],233] _ (230
],234] _ (231],235] _ (232
],236] _ (233],237] _ (234
],238] _ (235],239] _ (236
],240] _ (237],241] _ (238
],242] _ (239],243] _ (240
],244] _ (241],245] _ (242
],246] _ (243],247] _ (244
],248] _ (245],249] _ (246
],250] _ (247],251] _ (248
],252] _ (249],253] _ (250
],254] _ (251],255] _ (252
],256] _ (253],257] _ (254
],258] _ (255],259] _ (256
],260] _ (257],261] _ (258
],262] _ (259],263] _ (260
],264] _ (261],265] _ (262
],266] _ (263],267] _ (264
],268] _ (265],269] _ (266
],270] _ (267],271] _ (268
],272] _ (269],273] _ (270
],274] _ (271],275] _ (272
],276] _ (273],277] _ (274
],278] _ (275],279] _ (276
],280] _ (277],281] _ (278
],282] _ (279],283] _ (280
],284] _ (281],285] _ (282
],286] _ (283],287] _ (284
],288] _ (285],289] _ (286
],290] _ (287],291] _ (288
],292] _ (289],293] _ (290
],294] _ (291],295] _ (292
],296] _ (293],297] _ (294
],298] _ (295],299] _ (296
],300] _ (297],301] _ (298
],302] _ (299],303] _ (300
],304] _ (301],305] _ (302
],306] _ (303],307] _ (304
],308] _ (305],309] _ (306
],310] _ (307],311] _ (308
],312] _ (309],313] _ (310
],314] _ (311],315] _ (312
],316] _ (313],317] _ (314
],318] _ (315],319] _ (316
],320] _ (317],321] _ (318
],322] _ (319],323] _ (320
],324] _ (321],325] _ (322
],326] _ (323],327] _ (324
],328] _ (325],329] _ (326
],330] _ (327],331] _ (328
],332] _ (329],333] _ (330
],334] _ (331],335] _ (332
],336] _ (333],337] _ (334
],338] _ (335],339] _ (336
],340] _ (337],341] _ (338
],342] _ (339],343] _ (340
],344] _ (341],345] _ (342
],346] _ (343],347] _ (344
],348] _ (345],349] _ (346
],350] _ (347],351] _ (348
],352] _ (349],353] _ (350
],354] _ (351],355] _ (352
],356] _ (353],357] _ (354
],358] _ (355],359] _ (356
],360] _ (357],361] _ (358
],362] _ (359],363] _ (360
],364] _ (361],365] _ (362
],366] _ (363],367] _ (364
],368] _ (365],369] _ (366
],370] _ (367],371] _ (368
],372] _ (369],373] _ (370
],374] _ (371],375] _ (372
],376] _ (373],377] _ (374
],378] _ (375],379] _ (376
],380] _ (377],381] _ (378
],382] _ (379],383] _ (380
],384] _ (381],385] _ (382
],386] _ (383],387] _ (384
],388] _ (385],389] _ (386
],390] _ (387],391] _ (388
],392] _ (389],393] _ (390
],394] _ (391],395] _ (392
],396] _ (393],397] _ (394
],398] _ (395],399] _ (396
],400] _ (397],401] _ (398
],402] _ (399],403] _ (400
],404] _ (401],405] _ (402
],406] _ (403],407] _ (404
],408] _ (405],409] _ (406
],410] _ (407],411] _ (408
],412] _ (409],413] _ (410
],414] _ (411],415] _ (412
],416] _ (413],417] _ (414
],418] _ (415],419] _ (416
],420] _ (417],421] _ (418
],422] _ (419],423] _ (420
],424] _ (421],425] _ (422
],426] _ (423],427] _ (424
],428] _ (425],429] _ (426
],430] _ (427],431] _ (428
],432] _ (429],433] _ (430
],434] _ (431],435] _ (432
],436] _ (433],437] _ (434
],438] _ (435],439] _ (436
],440] _ (437],441] _ (438
],442] _ (439],443] _ (440
],444] _ (441],445] _ (442
],446] _ (443],447] _ (444
],448] _ (445],449] _ (446
],450] _ (447],451] _ (448
],452] _ (449],453] _ (450
],454] _ (451],455] _ (452
],456] _ (453],457] _ (454
],458] _ (455],459] _ (456
],460] _ (457],461] _ (458
],462] _ (459],463] _ (460
],464] _ (461],465] _ (462
],466] _ (463],467] _ (464
],468] _ (465],469] _ (466
],470] _ (467],471] _ (468
],472] _ (469],473] _ (470
],474] _ (471],475] _ (472
],476] _ (473],477] _ (474
],478] _ (475],479] _ (476
],480] _ (477],481] _ (478
],482] _ (479],483] _ (480
],484] _ (481],485] _ (482
],486] _ (483],487] _ (484
],488] _ (485],489] _ (486
],490] _ (487],491] _ (488
],492] _ (489],493] _ (490
],494] _ (491],495] _ (492
],496] _ (493],497] _ (494
],498] _ (495],499] _ (496
],500] _ (497],501] _ (498
],502] _ (499],503] _ (500
],504] _ (501],505] _ (502
],506] _ (503],507] _ (504
],508] _ (505],509] _ (506
],510] _ (507],511] _ (508
],512] _ (509],513] _ (510
],514] _ (511],515] _ (512
],516] _ (513],517] _ (514
],518] _ (515],519] _ (516
],520] _ (517],521] _ (518
],522] _ (519],523] _ (520
],524] _ (521],525] _ (522
],526] _ (523],527] _ (524
],528] _ (525],529] _ (526
],530] _ (527],531] _ (528
],532] _ (529],533] _ (530
],534] _ (531],535] _ (532
],536] _ (533],537] _ (534
],538] _ (535],539] _ (536
],540] _ (537],541] _ (538
],542] _ (539],543] _ (540
],544] _ (541],545] _ (542
],546] _ (543],547] _ (544
],548] _ (545],549] _ (546
],550] _ (547],551] _ (548
],552] _ (549],553] _ (550
],554] _ (551],555] _ (552
],556] _ (553],557] _ (554
],558] _ (555],559] _ (556
],560] _ (557],561] _ (558
],562] _ (559],563] _ (560
],564] _ (561],565] _ (562
],566] _ (563],567] _ (564
],568] _ (565],569] _ (566
],570] _ (567],571] _ (568
],572] _ (569],573] _ (570
],574] _ (571],575] _ (572
],576] _ (573],577] _ (574
],578] _ (575],579] _ (576
],580] _ (577],581] _ (578
],582] _ (579],583] _ (580
],584] _ (581],585] _ (582
],586] _ (583],587] _ (584
],588] _ (585],589] _ (586
],590] _ (587],591] _ (588
],592] _ (589],593] _ (590
],594] _ (591],595] _ (592
],596] _ (593],597] _ (594
],598] _ (595],599] _ (596
],600] _ (597],601] _ (598
],602] _ (599],603] _ (600
],604] _ (601],605] _ (602
],606] _ (603],607] _ (604
],608] _ (605],609] _ (606
],610] _ (607],611] _ (608
],612] _ (609],613] _ (610
],614] _ (611],615] _ (612
],616] _ (613],617] _ (614
],618] _ (615],619] _ (616
],620] _ (617],621] _ (618
],622] _ (619],623] _ (620
],624] _ (621],625] _ (622
],626] _ (623],627] _ (624
],628] _ (625],629] _ (626
],630] _ (627],631] _ (628
],632] _ (629],633] _ (630
],634] _ (631],635] _ (632
],636] _ (633],637] _ (634
],638] _ (635],639] _ (636
],640] _ (637],641] _ (638
],642] _ (639],643] _ (640
],644] _ (641],645] _ (642
],646] _ (643],647] _ (644
],648] _ (645],649] _ (646
],650] _ (647],651] _ (648
],652] _ (649],653] _ (650
],654] _ (651],655] _ (652
],656] _ (653],657] _ (654
],658] _ (655],659] _ (656
],660] _ (657],661] _ (658
],662] _ (659],663] _ (660
],664] _ (661],665] _ (662
],666] _ (663],667] _ (664
],668] _ (665],669] _ (666
],670] _ (667],671] _ (668
],672] _ (669],673] _ (670
],674] _ (671],675] _ (672
],676] _ (673],677] _ (674
],678] _ (675],679] _ (676
],680] _ (677],681] _ (678
],682] _ (679],683] _ (680
],684] _ (681],685] _ (682
],686] _ (683],687] _ (684
],688] _ (685],689] _ (686
],690] _ (687],691] _ (688
],692] _ (689],693] _ (690
],694] _ (691],695] _ (692
],696] _ (693],697] _ (694
],698] _ (695],699] _ (696
],700] _ (697],701] _ (698
],702] _ (699],703] _ (700
],704] _ (701],705] _ (702
],706] _ (703],707] _ (704
],708] _ (705],709] _ (706
],710] _ (707],711] _ (708
],712] _ (709],713] _ (710
],714] _ (711],715] _ (712
],716] _ (713],717] _ (714
],718] _ (715],719] _ (716
],720] _ (717],721] _ (718
],722] _ (719],723] _ (720
],724] _ (721],725] _ (722
],726] _ (723],727] _ (724
],728] _ (725],729] _ (726
],730] _ (727],731] _ (728
],732] _ (729],733] _ (730
],734] _ (731],735] _ (732
],736] _ (733],737] _ (734
],738] _ (735],739] _ (736
],740] _ (737],741] _ (738
],742] _ (739],743] _ (740
],744] _ (741],745] _ (742
],746] _ (743],747] _ (744
],748] _ (745],749] _ (746
],750] _ (747],751] _ (748
],752] _ (749],753] _ (750
],754] _ (751],755] _ (752
],756] _ (753],757] _ (754
],758] _ (755],759] _ (756
],760] _ (757],761] _ (758
],762] _ (759],763] _ (760
],764] _ (761],765] _ (762
],766] _ (763],767] _ (764
],768] _ (765],769] _ (766
],770] _ (767],771] _ (768
],772] _ (769],773] _ (770
],774] _ (771],775] _ (772
],776] _ (773],777] _ (774
],778] _ (775],779] _ (776
],780] _ (777],781] _ (778
],782] _ (779],783] _ (780
],784] _ (781],785] _ (782
],786] _ (783],787] _ (784
],788] _ (785],789] _ (786
],790] _ (787],791] _ (788
],792] _ (789],793] _ (790
],794] _ (791],795] _ (792
],796] _ (793],797] _ (794
],798] _ (795],799] _ (796
],800] _ (797],801] _ (798
],802] _ (799],803] _ (800
],804] _ (801],805] _ (802
],806] _ (803],807] _ (804
],808] _ (805],809] _ (806
],810] _ (807],811] _ (808
],812] _ (809],813] _ (810
],814] _ (811],815] _ (812
],816] _ (813],817] _ (814
],818] _ (815],819] _ (816
],820] _ (817],821] _ (818
],822] _ (819],823] _ (820
],824] _ (821],825] _ (822
],826] _ (823],827] _ (824
],828] _ (825],829] _ (826
],830] _ (827],831] _ (828
],832] _ (829],833] _ (830
],834] _ (831],835] _ (832
],836] _ (833],837] _ (834
],838] _ (835],839] _ (836
],840] _ (837],841] _ (838
],842] _ (839],843] _ (840
],844] _ (841],845] _ (842
],846] _ (843],847] _ (844
],848] _ (845],849] _ (846
],850] _ (847],851] _ (848
],852] _ (849],853] _ (850
],854] _ (851],855] _ (852
],856] _ (853],857] _ (854
],858] _ (855],859] _ (856
],860] _ (857],861] _ (858
],862] _ (859],863] _ (860
],864] _ (861],865] _ (862
],866] _ (863],867] _ (864
],868] _ (865],869] _ (866
],870] _ (867],871] _ (868
],872] _ (869],873] _ (870
],874] _ (871],875] _ (872
],876] _ (873],877] _ (874
],878] _ (875],879] _ (876
],880] _ (877],881] _ (878
],882] _ (879],883] _ (880
],884] _ (881],885] _ (882
],886] _ (883],887] _ (884
],888] _ (885],889] _ (886
],890] _ (887],891] _ (888
],892] _ (889],893] _ (890
],894] _ (891],895] _ (892
],896] _ (893],897] _ (894
],898] _ (895],899] _ (896
],900] _ (897],901] _ (898
],902] _ (899],903] _ (900
],904] _ (901],905] _ (902
],906] _ (903],907] _ (904
],908] _ (905],909] _ (906
],910] _ (907],911] _ (908
],912] _ (909],913] _ (910
],914] _ (911],915] _ (912
],916] _ (913],917] _ (914
],918] _ (915],919] _ (916
],920] _ (917],921] _ (918
],922] _ (919],923] _ (920
],924] _ (921],925] _ (922
],926] _ (923],927] _ (924
],928] _ (925],929] _ (926
],930] _ (927],931] _ (928
],932] _ (929],933] _ (930
],934] _ (931],935] _ (932
],936] _ (933],937] _ (934
],938] _ (935],939] _ (936
],940] _ (937],941] _ (938
],942] _ (939],943] _ (940
],944] _ (941],945] _ (942
],946] _ (943],947] _ (944
],948] _ (945],949] _ (946
],950] _ (947],951] _ (948
],952] _ (949],953] _ (950
],954] _ (951],955] _ (952
],956] _ (953],957] _ (954
],958] _ (955],959] _ (956
],960] _ (957],961] _ (958
],962] _ (959],963] _ (960
],964] _ (961],965] _ (962
],966] _ (963],967] _ (964
],968] _ (965],969] _ (966
],970] _ (967],971] _ (968
],972] _ (969],973] _ (970
],974] _ (971],975] _ (972
],976] _ (973],977] _ (974
],978] _ (975],979] _ (976
],980] _ (977],981] _ (978
],982] _ (979],983] _ (980
],984] _ (981],985] _ (982
],986] _ (983],987] _ (984
],988] _ (985],989] _ (986
],990] _ (987],991] _ (988
],992] _ (989],993] _ (990
],994] _ (991],995] _ (992
],996] _ (993],997] _ (994
],998] _ (995],999] _ (996
],1000] _ (997],1001] _ (998
],1002] _ (999],1003] _ (1000
],1004] _ (1001],1005] _ (1002
],1006] _ (1003],1007] _ (1004
],1008] _ (1005],1009] _ (1006
],1010] _ (1007],1011] _ (1008
],1012] _ (1009],1013] _ (1010
],1014] _ (1011],1015] _ (1012
],1016] _ (1013],1017] _ (1014
],1018] _ (1015],1019] _ (1016
],1020] _ (1017],1021] _ (1018
],1022] _ (1019],1023] _ (1020
],1024] _ (1021],1025] _ (1022
],1026] _ (1023],1027] _ (1024
],1028] _ (1025],1029] _ (1026
],1030] _ (1027],1031] _ (1028
],1032] _ (1029],1033] _ (1030
],1034] _ (1031],1035] _ (1032
],1036] _ (1033],1037] _ (1034
],1038] _ (1035],1039] _ (1036
],1040] _ (1037],1041] _ (1038
],1042] _ (1039],1043] _ (1040
],1044] _ (1041],1045] _ (1042
],1046] _ (1043],1047] _ (1044
],1048] _ (1045],1049] _ (1046
],1050] _ (1047],1051] _ (1048
],1052] _ (1049],1053] _ (1050
],1054] _ (1051],1055] _ (1052
],1056] _ (1053],1057] _ (1054
],1058] _ (1055],1059] _ (1056
],1060] _ (1057],1061] _ (1058
],1062] _ (1059],1063] _ (1060
],1064] _ (1061],1065] _ (1062
],1066] _ (1063],1067] _ (1064
],1068] _ (1065],1069] _ (1066
],1070] _ (1067],1071] _ (1068
],1072] _ (1069],1073] _ (1070
],1074] _ (1071],1075] _ (1072
],1076] _ (1073],1077] _ (1074
],1078] _ (1075],1079] _ (1076
],1080] _ (1077],1081] _ (1078
],1082] _ (1079],1083] _ (1080
],1084] _ (1081],1085] _ (1082
],1086] _ (1083],1087] _ (1084
],1088] _ (1085],1089] _ (1086
],1090] _ (1087],1091] _ (1088
],1092] _ (1089],1093] _ (1090
],1094] _ (1091],1095] _ (1092
],1096] _ (1093],1097] _ (1094
],1098] _ (1095],1099] _ (1096
],1100] _ (1097],1101] _ (1098
],1102] _ (1099],1103] _ (1100
],1104] _ (1101],1105] _ (1102
],1106] _ (1103],1107] _ (1104
],1108] _ (1105],1109] _ (1106
],1110] _ (1107],1111] _ (1108
],1112] _ (1109],1113] _ (1110
],1114] _ (1111],1115] _ (1112
],1116] _ (1113],1117] _ (1114
],1118] _ (1115],1119] _ (1116
],1120] _ (1117],1121] _ (1118
],1122] _ (1119],1123] _ (1120
],1124] _ (1121],1125] _ (1122
],1126] _ (1123],1127] _ (1124
],1128] _ (1125],1129] _ (1126
],1130] _ (1127],1131] _ (1128
],1132] _ (1129],1133] _ (1130
],1134] _ (1131],1135] _ (1132
],1136] _ (1133],1137] _ (1134
],1138] _ (1135],1139] _ (1136
],1140] _ (1137],1141] _ (1138
],1142] _ (1139],1143] _ (1140
],1144] _ (1141],1145] _ (114
```

Ex.: Some GNOME Project Guidelines

- “Programmers should strive to write good code so that it is easy to understand and modify by others
- Important qualities of good code
 - clarity
 - consistency
 - extensibility
 - correctness”

Example... (cont'd)

- “It is important to follow a good naming convention for the symbols in your programs
 - Function names should be of the form **module_submodule_operation**, for example, **gnome_canvas_set_scroll_region**
 - Symbols should have descriptive names: do not use **cntusr()**, use **count_active_users()** instead
 - Function names are lowercase, with underscores to separate words, like this:
gnome_canvas_set_scroll_region()”

Example... (cont'd)

- “Macros and enumerations are uppercase, with underscores to separate words, like this:
GNOMEUIINFO_SUBTREE() for a macro
- typedefs and structure names are mixed upper and lowercase, like this: **GnomeCanvasItem**, **GnomeIconList**
- Very short and terse names should only be used for the local variables of functions; never call a global variable **x**; use a longer name that tells what it does”

Another Ex.: Some Linux Guidelines

- “Tabs are 8 characters, and indentations too
- Put the opening brace last on the line, and put the closing brace first, thusly:

```
if (x is true) {  
    we do y  
}
```

- Functions have the opening brace at the beginning of the next line, thus:

```
int function(int x)  
{  
    body of function  
}
```

Our Guidelines! (These Matter!)

- File level comments
 - Author(s) name and unity id(s)
 - Brief purpose of program or module within program
- Function comments
 - Function's purpose
 - Inputs (global or parameters)
 - Outputs (return values and side effects)
 - Pre-conditions
 - Post-conditions (including side effects)

CSC230 - C and Software Tools © NC State University Computer Science Faculty



11

Our Guidelines! (These Matter!)

- Global Variables
 - Describe purpose
- Magic Numbers
 - Use #define except for obvious numbers (-1, 0, 1, 2)
 - Unless those numbers have a specific named purpose or are an exit code!!!

CSC230 - C and Software Tools © NC State University Computer Science Faculty



12

Our Guidelines! (These Matter!)

- Indentation
 - All indentation must be spaces (except for Makefiles)
 - The number of spaces for indentation must be consistent
 - 2 to 3 spaces
 - Indent:
 - Statements in a function
 - Statements in a control structure
 - Statements in a block { }

Our Guidelines! (These Matter!)

- Curly Braces
 - Functions – opening curly brace on next line
 - Everything else – opening curly brace at end of control structure
- Statements
 - 1 statement per line

Formatting with **indent**

- Many editors and IDEs (emacs, vim, Eclipse, Visual Studio, ...) automatically do formatting while you write your code
- Another option: use a standalone tool for formatting, e.g., **indent**
- Warning: **remove tabs** from your source code before using **indent**

Example: Code Before **indent**

```
#include <stdio.h>
#include <stdlib.h>

static int computelength (int, int);
int main (void) {
    typedef struct { int      left; int      right; int      length; }
        linesegment;
    linesegment *seg1, *seg2; seg1 = (linesegment *) malloc (sizeof (linesegment
)); seg2 = (linesegment *) malloc (sizeof (linesegment));
    (void) printf ("Enter left edge of segment 1: ");
    (void) scanf ("%d", &(seg1->left)); (void) printf ("Enter right edge of segment
1: "); (void) scanf ("%d", &(seg1->right)); (void) printf ("Enter left edge of s
egment 2: "); (void) scanf ("%d", &(seg2->left)); (void) printf ("Enter right ed
ge of segment 2: ");
    (void) scanf ("%d", &(seg2->right));
    seg1->length = computelength (seg1->left, seg1->right);
    seg2->length = computelength (seg2->left, seg2->right); if (seg1->length == seg2
->length) printf ("segment lengths are equal\n"); else printf ("segment lengths
are NOT equal\n"); return 0; } int computelength (int left, int right) { return
(right-left); }
```

- A mess!

Example: Using **indent**

indent prog.c

- Lots of options, customize to your preference
 - put these options in a file named **.indent.pro**, in your home directory
- Default indent does NOT meet all of our style guidelines!

```
static int computelength (int, int);
int
main (void)
{
    typedef struct {
        int     left;
        int     right;
        int     length;
    }          linesegment;
    linesegment *seg1,
                *seg2;

    seg1 = (linesegment *) malloc (sizeof (linesegment));
    seg2 = (linesegment *) malloc (sizeof (linesegment));
    (void) printf ("Enter left edge of segment 1: ");
    (void) scanf ("%d", &(seg1->left));
    (void) printf ("Enter right edge of segment 1: ");
    (void) scanf ("%d", &(seg1->right));
    (void) printf ("Enter left edge of segment 2: ");
    (void) scanf ("%d", &(seg2->left));
    (void) printf ("Enter right edge of segment 2: ");
    (void) scanf ("%d", &(seg2->right));
    seg1->length = computelength (seg1->left, seg1->right);
    seg2->length = computelength (seg2->left, seg2->right);
    if (seg1->length == seg2->length)
        printf ("segment lengths are equal\n");
    else
        printf ("segment lengths are NOT equal\n");
    return 0;
}

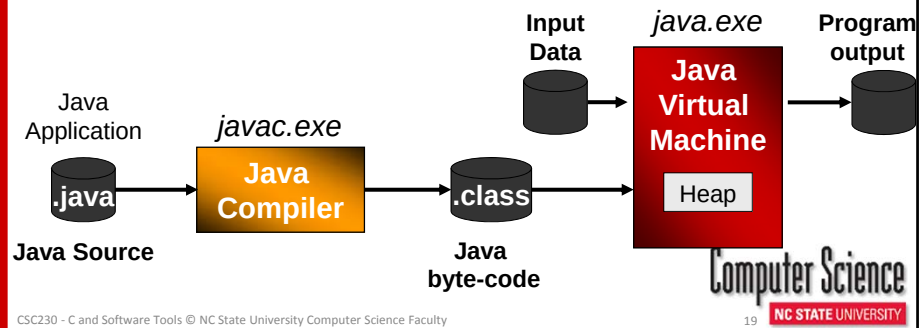
int
computelength (int left, int right) {
    return (right - left);
}
```

Example: after **indent**

- Much better!

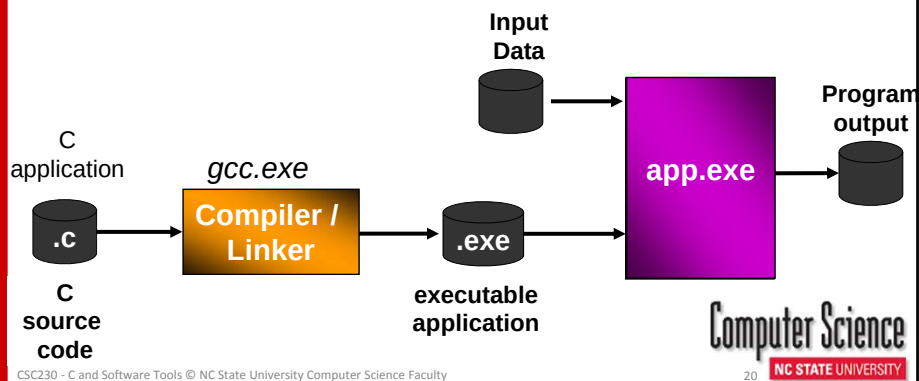
Executing Java Programs

1. Java source code is **compiled** into platform-independent intermediate form (*bytecode*)
2. This intermediate code is **interpreted** by the Java Virtual Machine (JVM)



Executing C Programs

1. HLL source code is **compiled** into the instruction set of the target computer
2. This code is loaded and **executed directly** by the host



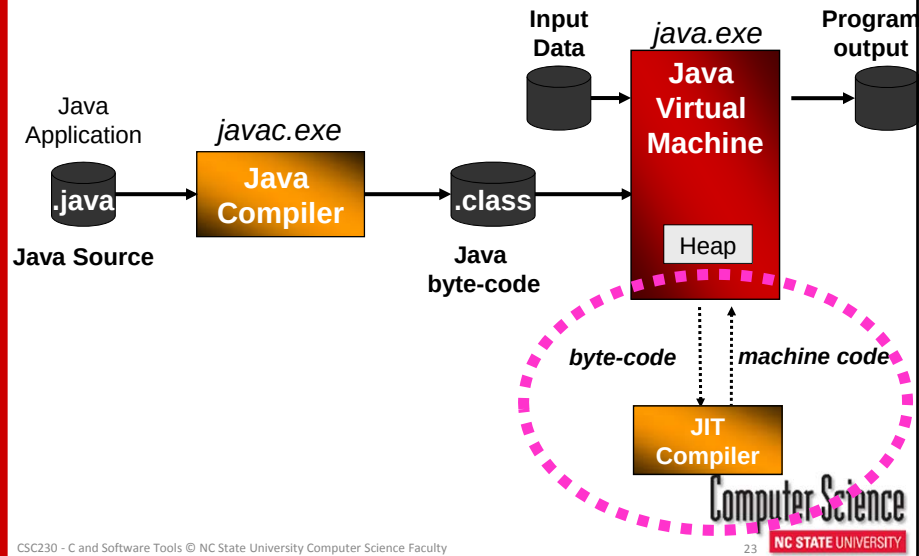
Platform Independence?

- Compiled
 - parts of the compiler (*front end*) are platform-independent
 - parts of the compiler (*back end*) are specific to the platform on which the program will be executed
- Interpreted
 - the Java compiler is platform-independent
 - the JVM is platform-specific

"Just-in-Time" Compiling

- Idea: compile a method to machine code just before first use
 - and reuse that machine code each time the method is invoked
- Benefits of interpreted + speed of compiled

JVM, Again



Comparison

Property	Better Compiled, or Interpreted?
Execution Speed	?
Error messages, debugging support	?
Platform Independence / Portability	?

- Another (major) benefit of interpreted languages: **dynamic typing of variables**
 - not supported in Java, however

Computer Science
NC STATE UNIVERSITY

• Source Code

Expanded Source Code	a=c+b*3;
----------------------	----------

Tokens	a = c + b * 3 ;
--------	-----------------

Parse Tree

```
graph TD
    unary-expression --> identifier
    unary-expression --> assignment-operator
    unary-expression --> assignment-expression
    identifier --> a
    assignment-operator --> equals
    assignment-expression --> 1
    assignment-expression --> plus
    assignment-expression --> 2
    assignment-expression --> asterisk
```

Computer Science
NC STATE UNIVERSITY

2

code generation

```
mov ebx, b
imul ebx, ebx, 3
mov ecx, c
```

Object Code

001110010111

Executable Code

0011100101110110101...

Computer Science

2

Using the **gcc** Compiler

- **gcc** is a high-quality, open source compiler available for most platforms
- At the command prompt, type

```
gcc -Wall -std=c99 <pgm.c>
```

where <pgm.c> is the C program source file

- Creates an executable **a.out**.
- **-std=c99** specifies that C99 standard features are allowed
- **-Wall** turns on all the important warning messages

CSC230 - C and Software Tools © NC State University Computer Science Faculty

Computer Science
NC STATE UNIVERSITY

27

Compiler... (cont'd)

- GNOME (and me): “Make sure your code compiles with absolutely no warnings from the compiler. These help you catch stupid bugs.”

CSC230 - C and Software Tools © NC State University Computer Science Faculty

Computer Science
NC STATE UNIVERSITY

28

Some Useful **gcc** Options

-c	Compile the source code but do not link (i.e., produce only the object file)
-E	Preprocess the source code only (i.e., expand macros, but do not compile the source code)
-o file	Put output in file named file
--version	Display version number of gcc
-std=c99	Support C99 language features
-Wall	Enable all warnings
-g	Produce information necessary to debug using gdb

CSC230 - C and Software Tools © NC State University Computer Science Faculty

29

NC STATE UNIVERSITY

Computer Science

gcc Options... (cont'd)

-O, -O1	Various optimization levels
-D name	Define name as a macro with value 1 (used for conditional compilation)
-llib	Search named library when linking
-I_{dir}	Add directory dir to the head of the list of directories to search for header files
-L_{dir}	Add directory dir to the list of directories to search for libraries containing object files (specified using the -l option)

CSC230 - C and Software Tools © NC State University Computer Science Faculty

30

NC STATE UNIVERSITY

Computer Science

A Word About C99

- The generations of C
 - K&R C
 - C89 (or C90)
 - C99
- } ISO standards
- We will use **C99** in this course
 - for the most part, C99 adds to / clarifies earlier versions, does not invalidate earlier code

(Some) Differences C89 ↔ C99

1. *Comments allowed to be C++ style (//)*
2. ***_Bool** macro is available*
3. *Additional library functions, and a few new header files*
4. *Variable length arrays*
5. *Variable declarations can appear anywhere in the code block*
6. *Variable declarations in **for** loops*
7. *Support for non-ASCII character sets (“wide” characters)*

(Some) Differences... (cont'd)

8. New **long long** integer data type
9. Functions must declare a return value
10. Macros may have variable number of arguments, denoted by ellipsis (...)
11. Functions may be inlined
12. Restricted pointers (prevent aliasing)

C99... (cont'd)

- gcc 4.4.6 supports most of C99, but you *may not be able to use...*
 - wide characters
 - variable length arrays
 - complex numbers
 - extended integer types (**long long**)

Console I/O in C

- I/O is provided by **standard library** functions
 - available on **all platforms**
- To use, your program must have

```
#include <stdio.h>
```
- ...and it doesn't hurt to also have

```
#include <stdlib.h>
```
- These are **preprocessor** statements; the *.h* files define function types, parameters, and constants from the standard library

Streams

- A **stream** is a **file** or a **device** from which data is read, and/or to which data is written
- By **default**, every C program automatically has 3 open streams, called
 - the **standard input**
 - the **standard output**
 - the **standard error**
- If you do not override them...
 - standard input = **the keyboard**
 - standard output & error = **the terminal window**

Streams... (cont'd)

- Note: the **EOF** character on your keyboard is either **ctrl-d** (Unix, Linux, Mac OS X) or **ctrl-z** (Windows)
- You can redirect the standard input from a file, e.g.,

```
pgm99 < infile.txt
```

- You can redirect the standard output to a file, e.g.,

```
pgm99 > outfile.txt
```

CSC230: C and Software Tools © NC State Computer Science Faculty

Computer Science
NC STATE UNIVERSITY

37

Reading One Character from Standard Input

- Definition (from `stdio.h`):

```
int getchar(void)
```

```
int c;
```

```
c = getchar();
```

```
if (c == EOF)
```

```
...
```

Notes

- **EOF** defined in `stdio.h`
- declaring `c` as type `char` and then comparing to **EOF** **may** fail ☹

CSC230: C and Software Tools © NC State Computer Science Faculty

Computer Science
NC STATE UNIVERSITY

38

Writing One Character to Standard Output

Definition (from `stdio.h`):

`int putchar(int c)`

```
char c;  
int b;  
...  
b = putchar((int) c);  
if (b == EOF)  
    ...
```

Program `echochar.c`

```
#include <stdio.h>  
  
int main ( void )  
{  
    int c;  
    c = getchar();  
    while (c != '\n') {  
        (void) putchar(c);  
        c = getchar();  
    }  
    putchar('\n');  
  
    return 0;  
}
```

Example: **echochar.c**

Demo...

- Keyboard input vs. input from a file
 - use editor to type the input in a **file** called **in.txt**
 - then run **echochar** with input redirected from the file
 - % **./echochar < in.txt**
- **No changes** to the program!

Demo...

The **printf()** function

- **putchar()** is too cumbersome to use for extensive, formatted output
- **printf()** is a much more convenient **library function** for formatted output, with built-in conversions of input parameters to printable form
- Def: **int printf(const char * format, ...)**
 - variable number of arguments
- **format** specifies how input arguments must be converted/formatted for output

Parts of **format**

1. **%** (mandatory)
2. 0 or more **flags** (infrequently used)
3. **Minimum output field width** (pad with spaces)
(useful for making things line up)
4. **.Precision** (minimum number of digits to right of decimal point)
(optional, default is 6 digits)
5. **type of format conversion** (mandatory)

Precision Matters

- **printf** the number 33.3:

Format Specifier	Output
%7.1f	33.3
%14.10f	33.3000000000
%.20f	33.29999999999999715783

Some Types of Conversions

Print as Type...	Specifier
char	%c
unsigned int	%u (in decimal) %o (in octal) %x, %X (in hex) (%lu, %lo, %lx for long)
signed int	%d, %i (in decimal) (%ld, %li for long)
float	%f
float	%e, %E (use scientific notation)
(string)	%s

CSC230: C and Software Tools © NC State Computer

45

Example

- Program

```
char c = 'a';
int i = 9999;
float f = 3.1415926535897932;

printf("c = %c (%o in octal)\n", c, c);
printf("i = %6d (%x in hex)\n", i, i);
printf("f = %8.5f (%e in sci. notation)\n",
      f, f);
```

Output:

```
c = a (141 in octal)
i =  9999 (270f in hex)
f =  3.14159 (3.141593e+00 in sci. notation)
```

CSC230: C and Software Tools © NC State Computer Science Faculty

Computer Science
NC STATE UNIVERSITY

46

Reminder

- Base 16 (“hex”):
 - $2F3_{16} = 2 * 16^2 + 15 * 16^1 + 3 = 755_{10}$
- Base 8 (“octal”):
 - $463_8 = 4 * 8^2 + 6 * 8^1 + 3 = 307_{10}$

Exercise 02.03: Basic I/O

- Write a program that
 - Reads 3 characters from standard input (all on one line, no spaces)
 - Outputs the characters in reverse order to standard output
- Make sure it compiles cleanly with the **-Wall** **-std=c99** options
- Make sure it is formatted cleanly and consistently
- Submit through Google Form