5

പ്രധാന ആശയങ്ങൾ

- ഡാറ്റ ഇനങ്ങൾ എന്ന ആശയം
- C++ ഡാറ്റ ഇനങ്ങൾ
- അടിസ്ഥാന ഡാറ്റ ഇനങ്ങൾ
- ടൈപ്പ് മോഡിഫയറുകൾ
- വേരിയബിളുകൾ
- ഓപ്പറേറ്ററുകൾ
  - അരിത്മറ്റിക്
  - റിലേഷണൽ
  - ലോജിക്കൽ
  - ഇൻപുട്ട്/ഔട്ട്പുട്ട്
  - അസൈൻമെന്റ്
- എക്സ്പ്രഷനുകൾ
  - അരിതമെറ്റിക്
  - റിലേഷണൽ
  - ലോജിക്കൽ
- പ്രസ്താവനകൾ
  - പ്രഖ്യാപനം
  - അസൈൻമെന്റ്
  - ഇൻപുട്ട്/ഔട്ട്പുട്ട്

T5J9E4

ഡാറ്റയുടെ ഇനങ്ങളും ഓപ്പറേറ്ററുകളും

ഇതിനു മുൻപുള്ള അധ്യായത്തിൽ C++ ന്റെ അടിസ്ഥാന നിർമ്മാണ ഘടകങ്ങളും പ്രോഗ്രാം വികസിപ്പിക്കുന്നതിനുപയോഗിക്കുന്ന IDE യും നാം പരിചയപ്പെട്ടു കഴിഞ്ഞു. കമ്പ്യൂട്ടറുകളിൽ നടക്കുന്ന പ്രധാന പ്രവർത്തനം ഡാറ്റ പ്രോസസ്സിംഗ് ആണെന്ന് നമുക്കറിയാമല്ലോ. എല്ലാ പ്രോഗ്രാമ്മിംഗ് ഭാഷകളും ഡാറ്റ കൈകാര്യം ചെയ്യുന്നതിന് പ്രാധാന്യം നൽകുന്നുണ്ട്. ചില പ്രത്യേകമായ സങ്കേതങ്ങൾ ഉപയോഗിച്ചുകൊണ്ടാണ് കമ്പ്യൂട്ടറുകളിൽ ഇൻപുട്ട് ചെയ്യപ്പെടുന്ന ഡാറ്റയുടെ ക്രമീകരണവും സംഭരണവും നടക്കുന്നത്. ഡാറ്റ സംഭരിക്കുന്നതിന് C++ന് മുൻകൂട്ടി നിർവചിച്ച ഒരു രൂപരേഖയുണ്ട്. സംഭരിക്കപ്പെട്ട ഡാറ്റ പിന്നീട് ഓപ്പറേറ്ററുകൾ ഉപയോഗിച്ച് പ്രോസസ്സ് ചെയ്യപ്പെടുന്നു. ഉപയോക്ത്യ നിർവചിത ഡാറ്റ ഇനങ്ങൾ എന്നു വിളിക്കുന്ന പുതിയ ഡാറ്റ ഇനങ്ങൾ നിർവചിക്കുന്നതിന് ഉപയോക്താവിനെ C++ അനുവദിക്കുന്നു.

ഇ++ഭാഷയുടെ മുഖ്യ ആശയങ്ങളായ ഡാറ്റ ഇനങ്ങൾ, ഓപ്പറേറ്ററുകൾ, പദപ്രയോഗങ്ങൾ, പ്രസ്താവനകൾ എന്നിവയെക്കുറിച്ച് നമുക്ക് ഈ അദ്ധ്യായത്തിൽ വിശദമായി പഠിക്കാം.

5.1 ഡാറ്റ ഇനങ്ങൾ എന്ന ആശയം (Concept of data types)

പരീക്ഷയ്ക്കുശേഷം ഒരു വിദ്യാർത്ഥിയുടെ പ്രോഗ്രസ് കാർഡ് തയ്യാറാക്കുന്നതിനായി നമുക്ക് അയാളുടെ രജിസ്റ്റർ നമ്പർ, റോൾ നമ്പർ, പേര്, വിലാസം, വിവിധ വിഷയങ്ങൾക്ക് ലഭിച്ച സ്കോർ, ഗ്രേഡുകൾ തുടങ്ങിയ ഡാറ്റ ആവശ്യമുണ്ട്. ഇത് കൂടാതെ, വിദ്യാർത്ഥിയുടെ സ്കോർ, ഹാജർ എന്നിവ ശതമാനത്തിൽ പ്രദർശിപ്പിക്കേണ്ടതുണ്ട്. ശാസ്ത്ര സംബന്ധിയായ ഡാറ്റ പ്രോസസിംഗ് പരിഗണിക്കുമ്പോൾ പ്രകാശത്തിന്റെ വേഗതയായ ($3x10^8 m/s$), ഗുരുത്വാകർഷണത്തിന്റെ വിലയായ ($9.8\ m/s^2$), ഇലക്ട്രോണിന്റെ ഇലക്ട്രിക് ചാർജായ ($-1.6x10^{-19}c$), തുടങ്ങിയ

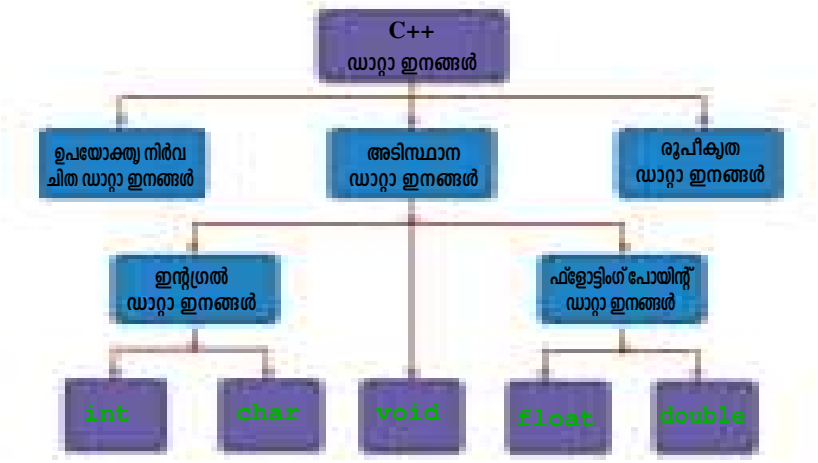
അക്കങ്ങളുടെ രൂപത്തിലുള്ള ഡാറ്റ ആവശ്യമായി വരാം.

മേൽ പറഞ്ഞതിൽ നിന്ന്, ക്യാരക്ടർ (character), ഇന്റിജർ (integer), ഭിന്ന സംഖ്യ (Real Number), വാക്കുകൾ (string) മുതലായ വിവിധതരം ഡാറ്റയുണ്ടെന്ന് നമുക്ക് അനുമാനിക്കാം. C++ലെ സാധുവായ ഒരു അക്ഷരം ഒരു ജോഡി ഏക ഉദ്ധരണികൾക്കുള്ളിൽ (single quotes) രേഖപ്പെടുത്തിയാൽ അത് ഒരു ക്യാരക്ടർ ഡാറ്റയെ പ്രതിനിധീകരിക്കുന്നു എന്ന് നാം കഴിഞ്ഞ അദ്ധ്യായത്തിൽ പഠിച്ചതാണല്ലോ. അതുപോലെ ദശാംശസ്ഥാനമില്ലാത്ത സംഖ്യകൾ ഇന്റിജർ (integer) ഡാറ്റയെ പ്രതിനിധാനം ചെയ്യുന്നു, ഭിന്നസംഖ്യകൾ ഫ്ളോട്ടിംഗ് പോയിന്റ് (floating point) ഡാറ്റ എന്നും, ഇരട്ട ഉദ്ധരണികളിൽ രേഖപ്പെടുത്തിയിരിക്കുന്നവ സ്ട്രിംഗ് (string) ഡാറ്റ എന്നും അറിയപ്പെടുന്നു. വിവിധ തരം ഡാറ്റ കൈകാര്യം ചെയ്യേണ്ടതിനാൽ എല്ലാ പ്രോഗ്രാമിംഗ് ഭാഷകളും അതിനുള്ള സംവിധാനം നൽകേണ്ടതാണ്. ഡാറ്റ ഇനങ്ങൾക്ക് പേരുകൾ നൽകികൊണ്ട് വിവിധതരം ഡാറ്റ കൈകാര്യം ചെയ്യുന്നതിനുള്ള സംവിധാനം C++ നൽകുന്നു. ഡാറ്റ ഇനങ്ങൾ (data types) എന്നാൽ ഡാറ്റയുടെ സ്വഭാവം, അതിൻമേൽ നടത്തുന്ന പ്രവർത്തനങ്ങൾ എന്നിവ തിരിച്ചറിയുന്നതിനുള്ള ഉപാധിയാണ്. ഈ സവിശേഷതകൾ വേർതിരിക്കുന്നതിനായി C++-ൽ വിവിധ ഡാറ്റ ഇനങ്ങൾ നിർവചിച്ചിരിക്കുന്നു.

അദ്ധ്യായം നാലിലെ അൽഗോരിതങ്ങളിൽ ഡാറ്റയെ സൂചിപ്പിക്കുവാൻ വേരിയബിളുകളാണ് നാം ഉപയോഗിച്ചത്. പ്രോഗ്രാമുകളിലും വേരിയബിളുകൾ തന്നെയാണ് ഡാറ്റയെ സൂചിപ്പിക്കുവാൻ ഉപയോഗിക്കുന്നത്. C++ ഭാഷയിൽ നാം പ്രോഗ്രാമുകൾ എഴുതുമ്പോൾ വേരിയബിളുകളെ അവ ഉപയോഗിക്കുന്നതിനുമുമ്പായി പ്രഖ്യാപിക്കേണ്ടതുണ്ട് (declaration of variable) ഈ വേരിയബിളുകൾ പ്രഖ്യാപിക്കുന്നതിന് ഡാറ്റ ഇനങ്ങൾ ആവശ്യമാണ്.

5.2 C++ ലെ ഡാറ്റ ഇനങ്ങൾ (C++ Data Types)

C++ വിവിധതരം ഡാറ്റാഇനങ്ങൾ കൊണ്ട് സമ്പുഷ്ടമാണ്. ഇവയെ സ്വഭാവം, വലിപ്പം, അവയുമായി ബന്ധപ്പെട്ട പ്രവർത്തനങ്ങൾ എന്നിവയുടെ അടിസ്ഥാനത്തിൽ ചിത്രം 6.1ൽ കാണുന്നതു പോലെ പലതായി തരം തിരിച്ചിട്ടുണ്ട്. ഡാറ്റ ഇനങ്ങളെ അടിസ്ഥാന ഡാറ്റ ഇനങ്ങൾ അല്ലെങ്കിൽ അന്തർ നിർമിത ഡാറ്റ ഇനങ്ങൾ (built-in data types), രൂപീകൃത ഡാറ്റ ഇനങ്ങൾ (Derived data types), ഉപയോക്ത നിർവചിത ഡാറ്റ ഇനങ്ങൾ (user defined data types) എന്നിങ്ങനെ തരം തിരിച്ചിരിക്കുന്നു.



ചിത്രം 5.1 : C++ ഡാറ്റാ ഇനങ്ങളുടെ തരംതിരിക്കൽ

അടിസ്ഥാന ഡാറ്റ ഇനങ്ങൾ: C++ കംപൈലറിൽ അടിസ്ഥാന ഡാറ്റ ഇനങ്ങൾ നിർവ്വചിച്ചിരിക്കുന്നു. അന്തർ നിർമിത ഡാറ്റ ഇനങ്ങൾ എന്നും അവ അറിയപ്പെടുന്നു. ഇവ വീണ്ടും വിഭജിക്കുവാൻ കഴിയാത്ത ഏറ്റവും ചെറിയ ഘടകങ്ങളാണ്. char, int, float, double, void എന്നിവയാണ് C++ലെ അഞ്ച് അടിസ്ഥാന ഡാറ്റ ഇനങ്ങൾ. ഇവയിൽ int, char, എന്നിവ പൂർണസംഖ്യാ ഡാറ്റ ഇനത്തിനു കീഴിൽ വരുന്നതാണ്. പൂർണ്ണ സംഖ്യകളെ മാത്രമേ ഇവയ്ക്ക് കൈകാര്യം ചെയ്യാൻ കഴിയുകയുള്ളൂ. ഭിന്നസംഖ്യകൾ സാധാരണയായി ദശാംശ സംഖ്യാ ഡാറ്റ ഇനം (floating point data type) എന്ന് അറിയപ്പെടുന്നു. ഇവയെ വ്യാപ്തിയുടെയും (range) കൃത്യതയുടെയും (presion) അടിസ്ഥാനത്തിൽ float, double എന്നിങ്ങനെ രണ്ടായി തരം തിരിച്ചിരിക്കുന്നു.

ഉപയോക്തൃ നിർവചിത ഡാറ്റ ഇനങ്ങൾ: പ്രോഗ്രാമർക്ക് സ്വന്തമായി ഡാറ്റ ഇനം നിർവചിക്കുവാനുള്ള സൗകര്യം C++ൽ ഉണ്ട്. സ്ട്രക്ചർ (struct), എന്യൂമറേഷൻ (enum), യൂണിയൻ (union), ക്ലാസ്സ് (class) തുടങ്ങിയവ ഇത്തരം ഡാറ്റ ഇനങ്ങൾക്കുള്ള ഉദാഹരണങ്ങളാണ്.

രൂപീകൃത ഡാറ്റ ഇനങ്ങൾ: അടിസ്ഥാന ഡാറ്റ ഇനങ്ങളെ ഗണങ്ങൾ ആക്കി മാറ്റിയോ വലിപ്പ മാറ്റം വരുത്തിയോ നിർമ്മിക്കപ്പെടുന്ന ഡാറ്റ ഇനങ്ങൾ രൂപീകൃത ഡാറ്റ ഇനങ്ങൾ എന്ന് അറിയപ്പെടുന്നു. അറെകൾ, പോയിന്ററുകൾ, ഫങ്ഷനുകൾ തുടങ്ങിയവ രൂപീകൃത ഡാറ്റ ഇനങ്ങൾക്കുള്ള ഉദാഹരണങ്ങളാണ്.

5.3 അടിസ്ഥാന ഡാറ്റ ഇനങ്ങൾ (fundamental data types):

അടിസ്ഥാന ഡാറ്റ ഇനങ്ങൾ മൗലികമായ സ്വഭാവ വിശേഷമുള്ളവയാണ്. അവയെ വീണ്ടും ചെറിയ ഭാഗങ്ങളായി വിഭജിക്കുവാൻ കഴിയില്ല. കമ്പൈലറിൽ നിർചിക്കപ്പെട്ടിരിക്കുതിനാൽ അവയുടെ വലിപ്പം (അനുവദിക്കപ്പെട്ട മെമ്മറിയുടെ അളവ്) കമ്പൈലറിനെ ആശ്രയിച്ചിരിക്കുന്നു. നാം ഉപയോഗിക്കുന്നത് GCC -ൽ ലഭ്യമായ കമ്പൈലർ ആയതിനാൽ ഡാറ്റയുടെ വലിപ്പവും അതുപോലെ ഡാറ്റയുടെ വ്യാപ്തിയും അതിന് അനുസൃതമായിരിക്കും. ടർബോ C++ IDE പോലുള്ള കംപയിലറുകൾ നിങ്ങൾ ഉപയോഗിക്കുമ്പോൾ ഇത് വ്യത്യസ്തമാകാം. അഞ്ചു അടിസ്ഥാന ഡാറ്റ ഇനങ്ങൾ താഴെ വിശദീകരിച്ചിരിക്കുന്നു.

int ഡാറ്റഇനം (പൂർണ സംഖ്യകൾക്കായി): ദശാംശ ഭാഗമില്ലാത്ത പൂർണ്ണ സംഖ്യകളാണ് ഇന്റിജറുകൾ. ഇവ പോസിറ്റീവോ, പൂജ്യമോ, നെഗറ്റീവോ ആകാം. ഒരു പ്രത്യേക പരിധിക്കുള്ളിലുള്ള ഇന്റിജറുകളെ പ്രതിനിധീകരിക്കാൻ ഉപയോഗിക്കുന്ന കീ വേർഡ് ആണ് int. int ഇനത്തിലുള്ള ഇന്റിജറുകൾക്ക് GCC നാല് ബൈറ്റ് മെമ്മറി അനുവദിക്കുന്നു. ആയതിനാൽ -2147483648 മുതൽ +2147483647 വരെയുള്ള സംഖ്യകളെ int ഡാറ്റാ ഇനം ഉപയോഗിച്ച് സൂചിപ്പിക്കാം. 69, 0, -112, 17, -32768, +32767 എന്നിവ int ഡാറ്റാ ഇനത്തിനുള്ള ഉദാഹരണങ്ങളാണ്. 22000000.00, - 2147483649 എന്നിവ അനുവദനീയമായ പരിധിക്ക് പുറത്തായതിനാൽ int ഡാറ്റാ ഇനമായി പരിഗണിക്കുകയില്ല.

char ഡാറ്റ ഇനം (ക്യാരക്ടർ സ്ഥിരാംഗങ്ങൾക്കുവേണ്ടി) : C++ ഭാഷയിലെ ക്യാരക്ടർ സെറ്റിലുൾപ്പെടുന്ന ചിഹ്നങ്ങൾ ആണ് ക്യാരക്ടറുകൾ. എല്ലാ അക്ഷരങ്ങൾ അക്കങ്ങൾ പ്രത്യേക ചിഹ്നങ്ങൾ വിരാമ ചിഹ്നങ്ങൾ തുടങ്ങിയവ ഈ വിഭാഗത്തിൽ ഉൾപ്പെടുന്നു.

ഈ ക്യാരക്ടറുകൾ ഡാറ്റയായി ഉപയോഗിക്കുമ്പോൾ അവയെ C++ ലെ char ഡാറ്റഇന മായി പരിഗണിക്കപ്പെടുന്നു. Char കീ വേർഡ് C++ ലെ ക്യാരക്ടർ ലിറ്ററലുകളെ പ്രതിനി ധീകരിക്കുന്നു എന്നു നമുക്ക് പറയാം.ഓരോ char ഇനത്തിലുള്ള ഡാറ്റായ്ക്കും 1 ബൈറ്റ് മെമ്മറി അനുവദിക്കുന്നു. 'a', '+', '/t', '0' മുതലായവ char ഡാറ്റാ ഇനത്തിൽപ്പെടുന്നവയാണ്.

char ഡാറ്റയെ ഇന്റിജർ ആയിട്ടാണ് പരിഗണിക്കുന്നത് എന്തുകൊണ്ടെന്നാൽ ASCII കോ ഡ് ഉപയോഗിച്ചാണ് കമ്പ്യൂട്ടർ ക്യാരക്ടറുകളെ തിരിച്ചറിയുന്നത്. മെമ്മറിയിൽ ക്യാരക്ടർ ഡാറ്റാ സംഭരിക്കുന്നത് അതിന്റേതായ ASCII കോഡ് ഉപയോഗിച്ചാണ്. ASCII കോഡ് ഇന്റിജറായതിനാലും അവ 8 ബിറ്റ്/1 ബൈറ്റ് സ്ഥലത്ത് സംഭരിക്കപ്പെടേണ്ടതിനാലും char ഡാറ്റാ ഇനത്തിന്റെ പരിധി -128 മുതൽ +127 വരെയാണ്.

float ഡാറ്റഇനം (ദശാംശ സംഖ്യകൾക്കായി): ദശാംശ ഭാഗത്തോടുകൂടിയ സംഖ്യ കളെ ഫ്ളോട്ടിംഗ് പോയിന്റ് സംഖ്യകളെന്നു വിളിക്കുന്നു. കമ്പ്യൂട്ടറിൽ ഫ്ളോട്ടിംഗ് പോയിന്റ് സംഖ്യകൾ രേഖപ്പെടുത്തുന്നത് ശാസ്ത്രീയമായ പ്രതീകങ്ങൾ ഉപയോഗിച്ചാ ണ്. ഉദാഹരണത്തിന് -47281.97 എന്ന് സംഖ്യയെ ശാസ്ത്രീയ പ്രതീകങ്ങൾ ഉപയോഗി ച്ചാണ് എഴുതുന്നത് 0.4728197×10^5 എന്നാണ്. ഇതിന്റെ ആദ്യഭാഗത്തെ (0.4728197) ഭിന്നാംശം (mantissa) എന്നും പത്തിന്റെ വർഗ്ഗമായ 5നെ (10^5) കൃത്യംങ്കം (exponent) എന്നും പറയുന്നു. ഫ്ളോട്ടിംഗ് പോയിന്റ് വിലകളെ പ്രതിനിധീകരിക്കാൻ കമ്പ്യൂട്ടർ സാധാ രണയായി കൃത്യങ്ക മാതൃക (E-notation) ഉപയോഗിക്കുന്നു. അത് പ്രകാരം 47281.97 എ ന്നത് 0.4728197E5 ആയാണ് രേഖപ്പെടുത്തുന്നത്. Eക്ക് മുമ്പുള്ള സംഖ്യ ഭിന്നാംശവും Eക്ക് ശേഷമുള്ള സംഖ്യ കൃത്യങ്കവുമാണ്. C++ൽ float എന്ന കീ വേർഡാണ് ഇത്തരം സംഖ്യകളെ സൂചിപ്പിക്കാൻ ഉപയോഗിക്കുന്നത്. float ഡാറ്റ ഇനത്തിൽപ്പെട്ട സംഖ്യകൾക്ക് 4 ബൈറ്റ് മെമ്മറി GCC അനുവദിക്കുന്നു. ഈ ഡാറ്റ തരത്തിലുള്ള സംഖ്യകൾക്ക് സാധാ രണയായി ദശാംശത്തിനുശേഷം 7 അക്കങ്ങൾ വരെ ആവാം.

Double ഡാറ്റ ഇനം (ഡബിൾ പ്രിസിഷൻ ദശാംശ സംഖ്യകൾക്കായി): ദശാംശത്തി നുശേഷം കൂടുതൽ അക്കങ്ങൾ വേണ്ട സംഖ്യകൾക്ക് അതായത് കൂടതുൽ കൃത്യത വേ ണ്ട ദശാംശ സംഖ്യകൾക്ക് ഉപയോഗിക്കുന്ന ഡാറ്റ ഇനമാണ് double. ഫ്ളോട്ട് ഡാറ്റാ ഇനം ഉപയോഗിച്ച് കൈകാര്യം ചെയ്യാവുന്ന സംഖ്യകളുടെ പരിധി ഈ ഡാറ്റാ ഇനം ഉപ യോഗിച്ച് വർധിപ്പിക്കാവുന്നതാണ്. എന്തുകൊണ്ടെന്നാൽ ഈ ഡാറ്റാ ഇനം ഫ്ളോട്ട് ഡാറ്റാ ഇനത്തെക്കാൾ ഇരട്ടി മെമ്മറി ഉപയോഗിക്കുന്നു. C++ ൽ double ന്റെ കൃത്യതയും പരിധിയും കുറഞ്ഞത് float ന്റെ അത്രയെങ്കിലും ആയിരിക്കണം. gcc ഇത്തരത്തിലുള്ള ഡാറ്റ സംഭരിക്കുന്നതിന് 8 ബൈറ്റ് മെമ്മറി നീക്കിവച്ചിരിക്കുന്നു. ഡബിൾ ഡാറ്റാ ഇന ത്തിൽ ദശാംശ സ്ഥാനത്തിനു ശേഷം 15 അക്കങ്ങൾ വരെ ആകാം.

void ഡാറ്റ തരം (എം.റ്റി. സെറ്റ് ഡാറ്റക്കായി): എംറ്റി സെറ്റിലെ ഡാറ്റയെ സൂചിപ്പിക്കാൻ ഉപയോഗിക്കുന്ന കീ വേഡാണ് വോയിഡ് (void). തീർച്ചയായും ഇതിന് മെമ്മറി ആവശ്യ മില്ല. ഈ ഡാറ്റാ ഇനത്തിന്റെ വിശദമായ ഉപയോഗം അധ്യായം 10 ൽ ചർച്ച ചെയ്യാം.

അടിസ്ഥാന ഡാറ്റ ഇനങ്ങളെ അവയുടെ വലിപ്പത്തിന്റെ അവരോഹണ ക്രമത്തിൽ double, float, int, char.void എന്ന് ക്രമീകരിക്കാം.

Name പേര്	Description വിശദീകരണം	Size വലിപ്പം	Range പരിധി
`char`	Character	1 byte	`signed`: -128 to 127 `unsigned`: 0 to 255
`int`	Integer	4 bytes	`signed`: -2147483648 to 2147483647 `unsigned`: 0 to 4294967295
`float`	Floating point number	4 bytes	$-3.4\times10^{+/-38}$ to $+3.4\times10^{+/-38}$ with approximately 7 significant digits
`double`	Double precision floating point number	8 bytes	$-1.7\times10^{+/-308}$ to $+1.7\times10^{+/-308}$ with approximately 15 significant digits
`void`	Null data	0 bytes	empty set

പട്ടിക 5.1: ഡാറ്റ ഇനങ്ങളും ടൈപ്പ് മോഡിഫയറുകളും

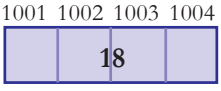
ഡാറ്റാഇനങ്ങൾ എങ്ങനെ വ്യത്യാസപ്പെട്ടിരിക്കുന്നു എന്ന് നിങ്ങൾക്ക് അറിയുന്നതിനുള്ള ഉദാഹരണങ്ങളാണ് ടേബിൾ 5.1 ൽ കൊടുത്തിരിക്കുന്നത്. ഇതിലുള്ള പല വിലകളും നിങ്ങളുടെ കമ്പ്യൂട്ടറിൽ വ്യത്യസ്തമായിരിക്കാം

5.4 വേരിയബിളുകൾ (variables)

ഡാറ്റ പരാമർശിക്കുന്നതിന് മെമ്മറിയിൽ അതിന്റെ സ്ഥാനങ്ങൾ തിരിച്ചറിയേണ്ടതുണ്ട്. മെമ്മറി സ്ഥാനങ്ങൾക്ക് നൽകുന്ന പേരുകളാണ് വേരിയബിളുകൾ. മെമ്മറി സ്ഥാനങ്ങളിൽ ഡാറ്റയെ സ്റ്റോർ ചെയ്യാനും വീണ്ടെടുക്കാനും ഉപയോഗിക്കുന്ന C++ -ലെ ഐഡന്റിഫയറുകളാണ് വേരിയബിളുകൾ. ഒരു വേരിയബിളിൽ സ്റ്റോർ ചെയ്തിട്ടുള്ള ഡാറ്റയുടെ സ്വഭാവവും അതിന്റെ വലിപ്പവും ആ വേരിയബിളിന്റെ ഡാറ്റ ഇനത്തിന് അനുസരിച്ചിരിക്കും. ഒരു വേരിയബിളിന് മൂന്ന് പ്രധാനപ്പെട്ട സ്വഭാവ സവിശേഷതകളുണ്ട്.

i. **വേരിയബളിന്റെ പേര് (variable name):** മെമ്മറിയിലെ ഒരു സ്ഥലത്ത് സംഭരിച്ചിരിക്കുന്ന ഡാറ്റ പരാമർശിക്കുന്നതിന് വേണ്ടി പ്രതീകാത്മകമായ ഉപയോഗിക്കുന്ന പേരാണിത്.

ii. **മെമ്മറി വിലാസം (memory address):** ഒരു ബൈറ്റു ഡാറ്റ വീതം സംഭരിക്കാൻ കഴിയുന്ന സെല്ലുകളുടെ (cell) ശേഖരമാണ് കമ്പ്യൂട്ടറിന്റെ RAM. RAM ലുള്ള ഓരോ സെല്ലും (ബൈറ്റ്) ഉപയോഗിക്കുന്നതിന് അവയ്ക്ക് തനതായ വിലാസങ്ങൾ നൽകപ്പെട്ടിരിക്കുന്നു. എല്ലാ വേരിയബിളുകളും RAM ലുള്ള ഒന്നോ അതിലധികമോ മെമ്മറി സ്ഥാനങ്ങളുമായി ബന്ധപ്പെട്ടിരിക്കുന്നു. അനുവദിച്ചിട്ടുള്ള മെമ്മറിയുടെ ആരംഭത്തിലെ സെല്ലിന്റെ വിലാസത്തെ പ്രാരംഭ വിലാസം (base address) എന്നു പറയുന്നു. സാധാരണഗതിയിൽ ഈ വിലാസം അനുവദിക്കുന്നത് കംപൈലർ ആണ്. ഈ വിലാസത്തെ വേരിയബിളിന്റെ എൽ മൂല്യം (L-Value) എന്നും വിളിക്കുന്നു. ചിത്രം 6.2 ൽ Num വേരിയബിളിന്റെ പ്രാരംഭ വിലാസം 1001 ആണ്.



ചിത്രം 5.2 : ഒരു വേരിയബിളിന്റെ മെമ്മറി പ്രതിനിധാനം

iii. **ഉള്ളടക്കം (Content):** ഒരു മെമ്മറി സ്ഥാനത്ത് സംഭരിച്ചിരിക്കുന്ന മൂല്യത്തെ വേരിയബിളിന്റെ ഉള്ളടക്കം എന്ന് വിളിയ്ക്കുന്നു. ഇതിനെ വേരിയബിളിന്റെ ആർ. മൂല്യം (R-value) എന്നും വിളിയ്ക്കുന്നു. ഉള്ളടക്കത്തിന്റെ തരവും വലിപ്പവും വേരിയബിളിന്റെ ഡാറ്റാ ഇനത്തെ ആശ്രയിച്ചിരിക്കുന്നു.

ചിത്രം 6.2 ഒരു വേരിയബിളിന്റെ മെമ്മറിയിലെ പ്രതിനിധാനം കാണിച്ചിരിക്കുന്നു. ഇവിടെ Num എന്നതു വേരിയബിളിന്റെ പേരും 1001, 1002, 1003, 1004 എന്നീ നാലു മെമ്മറി വിലാസങ്ങൾ ഉപയോഗിക്കുന്ന 4 ബൈറ്റ് മെമ്മറിയുമാണ്. ഈ വേരിയബിളിന്റെ ഉള്ളടക്കം 18 ആണ്. അതായത് Num ന്റെ L മൂല്യം. 1001 ഉം R മൂല്യം 18 ഉം ആണ്.

5.5. ഓപ്പറേറ്ററുകൾ (Operators):

കമ്പ്യൂട്ടറുകളിൽ പ്രവർത്തനങ്ങൾ (operations) നടപ്പിലാക്കുന്നതിന് പ്രേരിപ്പിക്കുന്ന മുൻകൂട്ടി നിശ്ചയിച്ചിട്ടുള്ള ചിഹ്നങ്ങളാണ് ഓപ്പറേറ്ററുകൾ. ഒരു ഓപ്പറേഷനിൽ പങ്കെടുക്കുന്നവയെ ഓപ്പറാൻഡ്സ് (operands) എന്നു വിളിക്കുന്നു. ഒരു ഓപ്പറന്റ് വേരിയയബിളോ സ്ഥിരാംഗമോ ആകാം.

ഉദാഹരണത്തിന് a+b എന്ന അരിത്മെറ്റിക് ഓപ്പറേഷനിൽ + (സങ്കലനം) ഓപ്പറേറ്ററും, a, b എന്നിവ ഓപ്പറാൻഡുകളും ആണ്. വിവിധ മാനദണ്ഡങ്ങൾക്കനുസൃതമായി C++ലെ ഓപ്പറേറ്ററുകളെ തരം തിരിച്ചിരിക്കുന്നു. C++ൽ ഓപ്പറേഷനുപയോഗിക്കുന്ന ഓപ്പറാൻഡ്കളുടെ എണ്ണം അനുസരിച്ച് ഓപ്പറേറ്ററുകളെ യൂനറി (unary), ബൈനറി (binary), ടെറിനറി (terinary) എിങ്ങനെ മൂന്നായി തരം തിരിച്ചിരിക്കുന്നു.

യൂനറി ഓപ്പറേറ്ററുകൾ (Unary Operators): ഒരു ഓപ്പറാന്റ് മാത്രമുള്ള പ്രവർത്തനങ്ങളിലെ ഓപ്പറേറ്ററുകളാണ് യൂനറി ഓപ്പറേറ്ററുകൾ. ഒരു സംഖ്യ പോസിറ്റീവ് അല്ലെങ്കിൽ നെഗറ്റീവ് എന്നു കാണിക്കാനുപയോഗിക്കുന്ന +, (-) ചിഹ്നങ്ങളാണ് സാധാരണ ഉപയോഗിക്കുന്ന യൂനറി ഓപ്പറേറ്ററുകൾ ചിഹ്നത്തോടു കൂടിയ ഒരു നമ്പറിന് മുൻപിൽ + ഓപ്പറേറ്റർ നൽകുമ്പോൾ നിലവിലുള്ള ചിഹ്നത്തിന് മാറ്റമുണ്ടാകുന്നില്ല. എന്നാൽ - നൽകുമ്പോൾ വിലയിൽ മാറ്റമുണ്ടാകുന്നു. ചിഹ്നത്തോടു കൂടിയ ഒരു സംഖ്യയിൽ യൂനറി ഓപ്പറേറ്റർ നാം പ്രയോഗിച്ചാൽ സംഖ്യയുടെ നിലവിലുള്ള ചിഹ്നം നേരേ വിപരീതമാകുന്നു. യൂനറി ഓപ്പറേറ്റിന്റെ ഉപയോഗം പട്ടിക 6.2ൽ കൊടുത്തിരിക്കുന്നു.

Variable x	Unary + +x	Unary- -x
8	8	-8
0	0	0
-9	-9	9

പട്ടിക 5.2 : യൂനറി ഓപ്പറേറ്ററുകൾ

ഇൻക്രിമന്റ് (increment) ++ (decriment) -- എന്നിവയും യൂനറി ഓപ്പറേറ്ററുകൾക്ക് ഉദാഹരണങ്ങളാണ്.

ബൈനറി ഓാപ്പറേറ്ററുകൾ (Binary Operator): ബൈനറി ഓപ്പറേറ്ററുകൾ രണ്ട് ഓപ്പറാൻഡുകളിൽ പ്രവർത്തിക്കുന്നു. അരിത്മെറ്റിക് ഓപ്പറേറ്ററുകൾ (arithmatic), റിലേഷണൽ ഓപ്പറേറ്ററുകൾ (relational), ലോജിക്കൽ ഓപ്പറേറ്ററുകൾ (logical) മുതലായവയാണ് സാധാരണയായി ഉപയോഗിക്കുന്ന ബൈനറി ഓപ്പറേറ്ററുകൾ.

ടെറിനറി ഓാപ്പറേറ്റർ (Terinary operator): ടെർനറി ഓപ്പറേറ്ററുകൾ മൂന്ന് ഓപ്പറാൻഡുകളിൽ പ്രവർത്തിക്കുന്നു. കണ്ടീഷണൽ (conditional) ഓപ്പറേറ്റർ (**?:**) ഇതിന് ഒരു ഉദാഹരണമാണ്.

മുകളിൽ പറഞ്ഞിരിക്കുന്ന ഓപ്പറേറ്ററുകളിൽ ചിലത് അടുത്ത ഭാഗങ്ങളിലും മറ്റു ചിലത് അധ്യായം ഏഴിലും ചർച്ച ചെയ്യാം.

പ്രവർത്തനരീതി അടിസ്ഥാനമാക്കി ഓപ്പറേറ്ററുകളെ അരിത്മാറ്റിക് (arithmatic), റിലേഷണൽ (relational), ലോജിക്കൽ (logical), ഇൻപുട്ട്/ ഔട്ട്പുട്ട് (input/output), അസൈൻമെന്റ് (assignment), ഷോർട്ട്-ഹാൻഡ് (short-hand), ഇൻക്രിമെന്റ് / ഡിക്രിമെന്റ് (increment/ decrement) എന്നിങ്ങനെ തരംതിരിച്ചിരിക്കുന്നു.

5.5.1 അരിത്മെറ്റിക് ഓപ്പറേറ്ററുകൾ (Arithmetic operators)

അടിസ്ഥാന ഗണിതപ്രക്രിയകളായ സങ്കലനം, വ്യവകലനം, ഗുണനം, ഹരണം എന്നിവയ്ക്ക് ഉപയോഗിക്കുന്ന ഓപ്പറേറ്ററുകളാണ് അരിത്മെറ്റിക് ഓപ്പറേറ്ററുകൾ. യഥാക്രമം +, ,* ,/എന്നീ ചിഹ്നങ്ങൾ ഇതിനായി ഉപയോഗിക്കുന്നു. ഹരണത്തിനു ശേഷമുള്ള ശിഷ്ടം ലഭിക്കുന്നതിനായി C++ ൽ മോഡുലസ് ഓപ്പറേറ്റർ (%) എന്നൊരു പ്രത്യേക ഓപ്പറേറ്ററും ഉപയോഗിക്കുന്നു. ഇവയെല്ലാം ബൈനറി ഓപ്പറേറ്ററുകളാണ്. + ഉം, - ഉം യുനറി ഓപ്പറേറ്റുകളായും ഉപയോഗിക്കുന്നു എന്നത് ശ്രദ്ധിക്കുക. ഈ പ്രവർത്തനങ്ങൾക്ക് സംഖ്യാസംബന്ധിയായ ഓപ്പറന്റുകളാണ് ആവശ്യമായിട്ടുള്ളത്. ഈ പ്രവർത്തനത്തിന് ശേഷം ലഭിക്കുന്ന ഫലവും ഒരു സംഖ്യയായിരിക്കും. പട്ടിക 6.3ൽ. ബൈനറി അരിത്മെറ്റിക് പ്രവർത്തനത്തിന്റെ ചില ഉദാഹരണങ്ങൾ കാണിച്ചിരിക്കുന്നു.

വേരിയബിൾ x	വേരിയബിൾ y	സങ്കലനം x + y	വ്യവകലനം x - y	ഗുണനം x * y	ഹരണം x / y
10	5	15	5	50	2
-11	3	-8	-14	-33	-3.66667
11	-3	8	14	-33	-3.66667
-50	-10	-60	-40	500	5

പട്ടിക 5.3 അരിത്മെറ്റിക് ഓപ്പറേറ്ററുകൾ

മോഡുലസ് ഓപ്പറേറ്ററുകൾ (Modulus operator (%)): മോഡ് അഥവാ മോഡുലസ് ഓപ്പറേറ്റർ ഹരണത്തിനുശേഷമുള്ള ശിഷ്ടം കണ്ടുപിടിക്കാൻ ഉപയോഗിക്കുന്നു. ഇത് ഇന്റിജർ ഓപ്പറാൻഡ്കുകളോടൊപ്പം മാത്രമേ ഉപയോഗിക്കാൻ കഴിയു. മോഡുലസ് പ്രക്രിയയുടെ ചില ഉദാഹരണങ്ങൾ പട്ടിക 6.4 ൽ കാണിച്ചിരിക്കുന്നു. ഈ പ്രക്രിയയുടെ ഫലത്തിന്റെ ചിഹ്നം ഒന്നാമത്തെ ഓപ്പറാൻഡിന്റെ ചിഹ്നം തന്നെ ആയിരിക്കുമെന്നത് ശ്രദ്ധിക്കുക. ഇവിടെ പട്ടികയിൽ ഒന്നാമത്തെ ഓപ്പറാന്റ് x ആണ്. ഉദാഹരണം പട്ടിക 5.4ൽ.

വേരിയബിൾ x	വേരിയബിൾ y	മോഡുലസ് ഓപ്പറേഷൻ x % y	വേരിയബിൾ x	വേരിയബിൾ y	മോഡുലസ് ഓപ്പറേഷൻ x % y
10	5	0	100	100	0
5	10	5	32	11	10
-5	11	-5	11	-5	1
5	-11	5	-11	5	-1
-11	-5	-1	-5	-11	-5

പട്ടിക 5.4: മോഡുലസ് ഓപ്പറേറ്റർ ഉപയോഗിച്ചുള്ള പ്രവർത്തനങ്ങൾ

സ്വയം വിലയിരുത്താം

1. അടിസ്ഥാന ഡാറ്റ ഇനങ്ങളെ ആരോഹണ ക്രമത്തിൽ ക്രമീകരിക്കുക.
2. ഒരു സംഭരണ സ്ഥാനത്തിനു നൽകുന്ന പേര് ..... എന്ന് അറിയപ്പെടുന്നു.
3. C++ ലെ ഒരു ടെർനറി ഓപ്പറേറ്ററുടെ പേരെഴുതുക.
4. x = -5, y = 3 ആയാൽ താഴെ കൊടുത്തിരിക്കുന്ന ഓപ്പറേഷനുകളുടെ ഔട്ട്പുട്ട് പ്രവചിക്കുക.

a) $-x$	c) $-x+-i$	e) $x \% -11$	g) $x \% y$
b) $-y$	d) $-x-y$	f) $x+y$	h) x/y
i) $x \times -y$	j) $-x \% -5$		

5.5.2 റിലേഷണൽ ഓഷറേറ്ററുകൾ (Relational Operators)

സംഖ്യ സംബന്ധിയായ ഡാറ്റയെ താരതമ്യം ചെയ്യുന്നതിനാണ് റിലേഷനൽ ഓപ്പറേറ്ററുകൾ ഉപയോഗിക്കുന്നത്. ഇവ ബൈനറി ഓപ്പറേറ്ററുകളാണ്. ഏതൊരു റിലേഷണൽ ഓപ്പറേഷന്റെയും ഫലം ശരി (true) അല്ലെങ്കിൽ തെറ്റ് (false) എന്നതായിരിക്കും. C++ൽ True നെ 1 കൊണ്ടും False നെ 0 കൊണ്ടും പ്രതിനിധീകരിക്കുന്നു. <(ചെറുതാണ്), < = (ചെറുതോ, തുല്യമോ ആണ്), > (വലുതാണ്), > = (വലുതോ, തുല്യമോ ആണ്), == (തുല്യമാണ്), ! = (തുല്യമല്ല). എന്നിങ്ങനെ 6 റിലേഷണൽ ഓപ്പറേറ്ററുകളാണ് C++ൽ ഉള്ളത്. തുല്യതാ പരിശോധനയ്ക്ക് രണ്ട് തുല്യ ചിഹ്നങ്ങൾ (=) ആവശ്യമാണെന്നത് ശ്രദ്ധിക്കുക. വിവിധ റിലേഷണൽ ഓപ്പറേറ്ററുകളുടെ ഉപയോഗവും അവയുടെ ഫലങ്ങളും പട്ടിക 6.5 ൽ കാണിച്ചിരിക്കുന്നു.

m	n	m<n	m>n	m<=n	m>=n	m!=n	m==n
12	5	0	1	0	1	1	0
-7	2	1	0	1	0	1	0
4	4	0	0	1	1	0	1

പട്ടിക 5.5 റിലേഷണൽ ഓപ്പറേറ്ററുകൾ ഉപയോഗിച്ചുള്ള പ്രവർത്തനങ്ങൾ

5.5.3 ലോജിക്കൽ ഓഷറേറ്ററുകൾ (Logical Operators)

റിലേഷണൽ ഓപ്പറേറ്ററുകൾ ഉപയോഗിച്ച് നമുക്ക് വിലകൾ താരതമ്യം ചെയ്യാം. ഉദാഹരണത്തിന് 3<5,num !=10 മുതലായവ C++ൽ ഇത്തരം താരതമ്യ പ്രവർത്തനങ്ങളെ റിലേഷണൽ പദപ്രയോഗങ്ങൾ എന്ന് വിളിക്കുന്നു. ചില സാഹചര്യങ്ങളിൽ രണ്ടോ അതിലധികമോ താരതമ്യങ്ങൾ സംയോജിപ്പിക്കേണ്ടതായി വരും. ഗണിതശാസ്ത്രത്തിൽ a>b>c എന്ന രീതിയിലുള്ള പദപ്രയോഗങ്ങൾ നമുക്ക് ഉപയോഗിക്കാം. എന്നാൽ C++ൽ ഇത് സാധ്യമല്ല. ഇവയെ a>b എന്നും b>c എന്നും വേർതിരിച്ച് & എന്ന ലോജിക്കൽ ഓപ്പറേറ്റർ ഉപയോഗിച്ച് സംയോജിപ്പിക്കുന്നു. അതായത് (a>b) && (b>c). ഇത്തരം ലോജിക്കൽ സംയോഗങ്ങളുടെ ഫലവും (true) (1) അല്ലെങ്കിൽ (false) (0) ആയിരിക്കും. && (ലോജിക്കൽ ആൻഡ് (AND),! (ലോജിക്കൽ ഓർ (OR)), ! (ലോജിക്കൽ നോട്ട് (NOT)) എന്നിവയാണ് C++ ലെ ലോജിക്കൽ ഓപ്പറേറ്ററുകൾ.

ലോജിക്കൽ ആന്റ് (logical AND) ഓപ്പറേറ്റർ: E1, E2 എന്നീ രണ്ട് റിലേഷൻ പദപ്രയോഗങ്ങൾ logical AND ഉപയോഗിച്ച് സംയോജിപ്പിക്കുമ്പോൾ ഫലം true(1) ലഭിക്കണമെങ്കിൽ E1, E2 എന്നിവ രണ്ടും true(1) തന്നെ ആയിരിക്കണം. അല്ലാത്ത എല്ലാ സന്ദർഭങ്ങളിലും ഫലം false(0) ആയിരിക്കും. വിവിധ ഇൻപുട്ടുകൾക്ക് അനുസരിച്ചുള്ള ലോജിക്കൽ AND പ്രക്രിയയുടെ ഫലം പട്ടിക 6.6 ൽ കാണിച്ചിരിക്കുന്നു.

E1	E2	E1&&E2
0	0	0
0	1	0
1	0	0
1	1	1

പട്ടിക 5.6ആന്റ് ഓപ്പറേറ്ററിന്റെ ഉപയോഗം

ഉദാഹരണം 10>5&&15<25 ഫലം true (1). 10>5 &&100<25 ഫലം false (0).

ലോജിക്കൽ ഓർ (logical OR) ഓപ്പറേറ്റർ: E1, E2 എന്നീ രണ്ട് റിലേഷൻ പദപ്രയോഗങ്ങൾ logical OR ഉപയോഗിച്ച് സംയോജിപ്പിക്കുമ്പോൾ ഫലം false(0) ലഭിക്കുമെങ്കിൽ E1, E2 എന്നിവ രണ്ടും false (0) ആയിരിക്കണം. അല്ലാത്ത എല്ലാ സന്ദർഭങ്ങളിലും ഫലം true(1) ആയിരിക്കും. വിവിധ ഇൻപുട്ടുകൾക്ക് അനുസരിച്ചുള്ള ലോജിക്കൽ OR പ്രക്രിയയുടെ ഫലം പട്ടിക 6.7ൽ കാണിച്ചിരിയ്ക്കുന്നു.

E1	E2	E1||E2
0	0	0
0	1	1
1	0	1
1	1	1

പട്ടിക 5.7 ഓർ ഓപ്പറേറ്ററിന്റെ ഉപയോഗം

ഉദാഹരണം: 10>15 ||100<25 ഫലം true(1), 10>15 ||100<90 ഫലം false (0).

ലോജിക്കൽ നോട്ട് (logical NOT) ഓപ്പറേറ്റർ: റിലേഷനൽ പദപ്രയോഗങ്ങളുടെ ഫലം വിപരീതമാക്കാനാണ് logical NOT ഉപയോഗിക്കുന്നത്. ഇത് ഒരു യൂനറി ഓപ്പറേഷനാണ്.

വിവിധ ഇൻപുട്ടുകൾക്ക് അനുസരിച്ചുള്ള ലോജിക്കൽ NOT പദപ്രയോഗത്തിന്റെ ഫലം പട്ടിക 6.8ൽ കാണിച്ചിരിക്കുന്നു.

E1	!E1
0	1
1	0

പട്ടിക 5.8 നോട്ട് ഓപ്പറേറ്ററിന്റെ ഉപയോഗം

ഉദാഹരണം ! (100<2) ഫലം 1

! (100>2) ഫലം 0 (False)

5.5.4 ഇൻപുട്ട് / ഔട്ട്പുട്ട് ഓപ്പറേറ്ററുകൾ (Input/Output Operators)

ഇൻപുട്ട് പ്രവർത്തനങ്ങൾക്ക് സാധാരണയായി ഉപയോക്താവിന്റെ ഇടപെടൽ ആവശ്യമാണ്. ഇൻപുട്ട് പ്രോസസ്സിൽ കീബോഡ് വഴി നൽകുന്ന ഡാറ്റ മെമ്മറി ലൊക്കേഷനുകളിൽ സൂക്ഷിക്കുന്നു. C++ൽ ഇൻപുട്ട് ഓപ്പറേഷൻ ചെയ്യുന്നതിനായി >> ഓപ്പറേറ്റർ ഉപയോഗിക്കുന്നു. ഈ ഓപ്പറേറ്റർ ഗേറ്റ് ഫ്രം (get from) അഥവാ എക്സ്ട്രാക്ഷൻ (extraction) ഓപ്പറേറ്റർ എന്ന് അറിയപ്പെടുന്നു. രണ്ട് > ചിഹ്നങ്ങൾ ഉപയോഗിച്ചാണ് ഈ ചിഹ്നം നിർമ്മിച്ചിരിക്കുന്നത്.

ഇതുപോലെ ഔട്ട്പുട്ട് പ്രവർത്തനങ്ങളിൽ ഡാറ്റ റാമിൽ നിന്നും ഔട്ട്പുട്ട് ഉപകരണത്തിലേക്ക് മാറ്റുന്നു. സാധാരണയായി ഫലം നേരിട്ട് ലഭിക്കാൻ ഉപയോഗിക്കുന്ന ഔട്ട്പുട്ട് ഉപകരണം മോണിറ്ററാണ്. പുട്ട് ടു (Put to) അഥവാ ഇൻസേർഷൻ (insertion) ഓപ്പറേറ്റർ എന്നു വിളിക്കുന്ന <<ഓപ്പറേറ്റർ ഔട്ട്പുട്ട് പ്രവർത്തനത്തിന് ഉപയോഗിക്കുന്നു. ഇത് രണ്ട് <ചിഹ്നങ്ങൾ ഉപയോഗിച്ചാണ് നിർമ്മിച്ചിരിക്കുന്നത്.

5.5.5 അസൈസ്മെന്റ് ഓപ്പറേറ്റർ (=) (Assignment operator (=)

സാധാരണയായി ഒരു വില മെമ്മറിയിൽ സംഭരിക്കുന്നതിനായി വിലനൽകൽ ഓപ്പറേറ്റർ ഉപയോഗിക്കുന്നു. ഇത് ഒരു ബൈനറി ഓപ്പറേറ്ററായതിനാൽ ഇവയ്ക്ക് രണ്ട് ഓപ്പറാന്റുകൾ ആവശ്യമാണ്. ആദ്യത്തെ ഓപ്പറാന്റ് ഒരു വേരിയബിൾ ആയിരിക്കണം. അതിലാണ് രണ്ടാമത്തെ ഓപ്പറാന്റിന്റെ മൂല്യം സൂക്ഷിക്കുന്നത്.

ഇനം	വിശദീകരണം
a=b	വേരിയബിൾ **b** യുടെ വില **a** ൽ സംഭരിക്കുന്നു
a=3	സ്ഥിരാംഗം **3** വേരിയബിൾ **a** ൽ സംഭരിക്കുന്നു

പട്ടിക 5.9 അസൈൻമെന്റ് ഓപ്പറേറ്റർ

ചില ഉദാഹരണങ്ങൾ പട്ടിക 5.9ൽ കാണിച്ചിരിക്കുന്നു.

റിലേഷണൽ ഓപ്പറേറ്ററായ = = ഉപയോഗത്തെപ്പറ്റി ഭാഗം 6.6.2ൽ നമ്മൾ ചർച്ച ചെയ്തിരുന്നു. ഈ രണ്ടു ഓപ്പറേറ്ററുകൾ തമ്മിലുള്ള വ്യത്യാസം ശ്രദ്ധിക്കുക. = ചിഹ്നം ഒരു വേരിയബിളിനു വില നൽകുന്നതിനും എന്നാൽ = = ചിഹ്നം രണ്ട് വിലകളെ തമ്മിൽ താരതമ്യം ചെയ്ത് true അല്ലെങ്കിൽ false എന്ന ഉത്തരം നൽകുന്നതിനും ഉപയോഗിക്കുന്നു.

5.6. പ്രയോഗങ്ങൾ (expressions)

ഒരു പദപ്രയോഗം ഓപ്പറേറ്ററുകളും ഓപ്പറാന്റുകളും ചേർന്നതാണ്. ഓപ്പറാന്റുകൾ സ്ഥിരാംഗങ്ങളോ വേരിയബിളുകളോ ആകാം. എല്ലാ പദപ്രയോഗങ്ങളും പൂർത്തീകരിച്ചതിനുശേഷമേ ആ പ്രയോഗത്തിന്റെ അന്തിമ ഫലം ലഭ്യമാകൂ. ഈ ഫലം പദപ്രയോഗം തിരികെ നൽകിയ വില എന്ന് അറിയപ്പെടുന്നു. ഉപയോഗിച്ചിരിക്കുന്ന ഓപ്പറേറ്ററുകളുടെ അടിസ്ഥാനത്തിൽ പദപ്രയോഗങ്ങളെ പ്രധാനമായും അരിത്മാറ്റിക് പദപ്രയോഗങ്ങൾ, റിലേഷണൽ പദപ്രയോഗങ്ങൾ, ലോജിക്കൽ പദപ്രയോഗങ്ങൾ എന്നിങ്ങനെ തരംതിരിച്ചിരിക്കുന്നു.

5.6.1 അരിത്മാറ്റിക് പ്രയോഗങ്ങൾ (arithmetic expressions)

അരിത്മാറ്റിക് ഓപ്പറേറ്ററുകൾ മാത്രം ഉപയോഗിച്ചിട്ടുള്ള പദപ്രയോഗങ്ങളെ അരിത്മാറ്റിക് പദപ്രയോഗങ്ങൾ എന്ന് വിളിക്കുന്നു. ഇവിടെ ഓപ്പറാന്റുകൾ സംഖ്യകളാണ്. അവ വേരിയബിളുകളോ സ്ഥിരാംഗങ്ങളോ ആകാം. ഈ പദപ്രയോഗത്തിൽ നിന്നും ലഭ്യമാകുന്ന വിലയും ഒരു സംഖ്യ ആയിരിക്കും. അരിത്മാറ്റിക് പദപ്രയോഗങ്ങളെ വീണ്ടും പൂർണ്ണ സംഖ്യാപദപ്രയോഗങ്ങൾ, ദശാംശസംഖ്യാ (real) പദപ്രയോഗങ്ങൾ, സ്ഥിരാംഗ പദപ്രയോഗങ്ങൾ എന്നിങ്ങനെ തരം തിരിച്ചിരിക്കുന്നു.

പൂർണ്ണസംഖ്യാ പ്രയോഗങ്ങൾ: ഒരു അരിത്മെറ്റിക് പദപ്രയോഗത്തിൽ പൂർണ്ണസംഖ്യകൾ മാത്രമേ ഉൾകൊള്ളുന്നുള്ളൂ എങ്കിൽ അതിനെ പൂർണ്ണസംഖ്യാപദപ്രയോഗം എന്ന് വിളിക്കുന്നു. ഇവയുടെ ഫലവും ഒരു പൂർണ്ണസംഖ്യ ആയിരിക്കും.

ഉദാഹരണത്തിന്: x, y എന്നിവ പൂർണ്ണസംഖ്യാ വേരിയബിളുകൾ ആണെങ്കിൽ ചില പൂർണ്ണ സംഖ്യാ പദപ്രയോഗവും അവയുടെ ഫലങ്ങളും പട്ടിക 5.10 ൽ കൊടുത്തിരിക്കുന്നു. എല്ലാ പദപ്രയോഗങ്ങളുടെയും ഫലം ഒരു പൂർണ്ണസംഖ്യ ആയിരിക്കും എന്നത് ശ്രദ്ധിക്കുക.

x	y	x + y	x / y	-x + x * y	5 + x / y	x % y
5	2	7	2	5	7	1
6	3	9	2	12	7	0

പട്ടിക 5.10 പൂർണ്ണ സംഖ്യാ പ്രയോഗങ്ങളും അവയുടെ ഫലങ്ങളും

ഫ്ളോട്ടിംഗ് പോയിന്റ് പ്രയോഗങ്ങൾ (floating point/ real expression): ഒരു അരിത്മാറ്റിക് പദപ്രയോഗത്തിൽ എല്ലാ വിലകളും ദശാംശസഖ്യകൾ ആണെങ്കിൽ അവയെ ദശാംശസഖ്യ അഥവാ ഭിന്ന സംഖ്യാപദപ്രയോഗം എന്ന് വിളിക്കുന്നു. ഇതിന്റെ ഫലം തീർച്ചയായും ഒരു ദശാംശസംഖ്യ ആയിരിക്കും. x, y എന്നിവ ദശാംശസംഖ്യാ വേരിയബിൾ ആണെന്ന് കരുതുക. ചില ദശാംശസംഖ്യാപദപ്രയോഗങ്ങളും അവയുടെ ഫലങ്ങളും പട്ടിക 5.11 ൽ കൊടുത്തിരിക്കുന്നു.

x	y	x + y	x / y	-x + x * y	5 + x / y	x * x / y
5.0	2.0	7.0	2.5	5.0	7.5	12.5
6.0	3.0	9.0	2.0	12.0	7.0	12.0

പട്ടിക 5.11: ഫ്ളോട്ടിംഗ് പോയിന്റ് സംഖ്യാ പ്രയോഗങ്ങളും അവയുടെ ഫലങ്ങളും

മുകളിൽ കൊടുത്തിരിക്കുന്ന എല്ലാ പദപ്രയോഗങ്ങളുടെയും ഉത്തരം ദശാംശസംഖ്യകളാണ് എന്ന് കാണാൻ കഴിയും.

ഒരു അരിത്മാറ്റിക് പദപ്രയോഗത്തിൽ ഉപയോഗിക്കുന്ന എല്ലാ ഓപ്പറാന്റുകളും സ്ഥിരാംഗങ്ങളാണെങ്കിൽ അതിനെ സ്ഥിരാംഗപദപ്രയോഗം (const. expression) എന്നു വിളിക്കുന്നു. ഉദാ: 20+5/2.0. സ്ഥിരാംഗങ്ങളായ 15,3.14, 'a' എന്നിവയും സ്ഥിരാംഗപദപ്രയോഗങ്ങളായി അറിയപ്പെടുന്നു.

5.6.2 റിലേഷണൽ പ്രയോഗങ്ങൾ (relational expressions)

റിലേഷണൽ ഓപ്പറേറ്ററുകൾ ഉപയോഗിക്കുന്ന പദപ്രയോഗങ്ങളെ റിലേഷണൽ പദപ്രയോഗങ്ങൾ എന്ന് വിളിക്കുന്നു. ഇവ true(1) അല്ലെങ്കിൽ false(0) എന്ന ഫലം നൽകുന്നു. ഇത്തരം പദപ്രയോഗങ്ങളിൽ ഓപ്പറാന്ററുകളായി സംഖ്യകളാണ് ഉപയോഗിക്കുക. ഇവയുടെ ചില ഉദാഹരണം പട്ടിക 5.15 ൽ കാണിച്ചിരിക്കുന്നു.

x	y	x > y	x == y	x+y !=y	x-2 == y+1	x*y == 6*y
5.0	2.0	1 (True)	0 (False)	1 (True)	1 (True)	0 (False)
6	13	0 (False)	0 (False)	1 (True)	0 (False)	1 (True)

പട്ടിക 5.12 റിലേഷണൽ പ്രയോഗങ്ങളും അവയുടെ ഫലങ്ങളും

അരിത്മാറ്റിക് ഓപ്പറേറ്ററുകൾക്ക് റിലേഷണൽ ഓപ്പറേറ്ററേക്കാൾ മുൻഗണയുണ്ടെന്ന്നമുക്കറിയാം. ഒരു റിലേഷണൽ ഓപ്പറേറ്ററിന്റെ ഇരുവശങ്ങളിലായി അരിത്മാറ്റിക് പദപ്രയോഗങ്ങൾ ഉപയോഗിക്കുമ്പോൾ ആദ്യം അരിത്മാറ്റിക് ഓപ്പറേഷനുകൾ ചെയ്യുകയും അതിന് ശേഷം ആ ഫലങ്ങൾ താരതമ്യം ചെയ്യുന്നു. പട്ടികയിലെ ചില പദപ്രയോഗങ്ങളിൽ

അരിത്മാറ്റിക് ഓപ്പറേറ്ററും റിലേഷണൽ ഓപ്പറേറ്ററുകളും ഉൾപ്പെട്ടിരിക്കുന്നു. വിവിധ തരം ഓപ്പറേറ്ററുകൾ ഉൾപ്പെട്ടിട്ടുണ്ടെങ്കിലും ഇവയുടെ ഫലം true(1) അല്ലെങ്കിൽ false (0) ആയതിനാൽ അവയെ റിലേഷണൽ പദപ്രയോഗങ്ങൾ എന്നാണ് വിളിക്കുന്നത്.

5.6.3 ലോജിക്കൽ പ്രയോഗങ്ങൾ (logical expressions)

രണ്ടോ അതിലധികമോ റിലേഷണൽ പദപ്രയോഗങ്ങളെ ലോജിക്കൽ ഓപ്പറേറ്റർ ഉപയോഗിച്ച് ലോജിക്കൽ പദപ്രയോഗങ്ങൾ സംയോജിപ്പിക്കുന്നു. ഇവയുടെ ഫലം true(1) അല്ലെങ്കിൽ false (0) എന്നായിരിക്കും. ലോജിക്കൽ പദപ്രയോഗത്തിൽ വേരിയബിളുകൾ, സ്ഥിരാംഗങ്ങൾ ലോജിക്കൽ ഓപ്പറേറ്ററുകൾ, റിലേഷനൽ ഓപ്പറേറ്ററുകൾ എന്നിവ ഉൾപ്പെടാവുന്നതാണ്. ചില ഉദാഹരണങ്ങൾ പട്ടിക 5.13 ൽ കാണിച്ചിരിക്കുന്നു.

x	y	x>=y && x==20	x==5||y==0	x==y && y+2==0	!(x==y)
5.0	2.0	0 (False)	1 (True)	0 (False)	1 (True)
20	13	1 (True)	0 (False)	0 (False)	1 (True)

പട്ടിക 5.13 ലോജിക്കൽ പ്രയോഗങ്ങളും അവയുടെ ഫലങ്ങളും

പട്ടിക 5.13 ൽ കാണുന്നതു പോലെ ചില പദപ്രയോഗങ്ങളിൽ ലോജിക്കൽ ഓപ്പറേറ്ററുകളെ കൂടാതെ അരിതമാറ്റിക് ഓപ്പറേറ്ററുകളും റിലേഷൺ ഓപ്പറേറ്ററുകളും ഉൾപ്പെട്ടിട്ടുണ്ടെങ്കിലും ഈ പ്രയോഗങ്ങളെ ലോജിക്കൽ പ്രയോഗങ്ങളായി കണക്കാക്കുന്നു. അവസാനം ചെയ്യുന്ന പ്രവർത്തനം ലോജിക്കൽ പ്രവർത്തനം ആയതിനാലും അതിന്റെ ഫലം True അല്ലെങ്കിൽ False ആയത് കൊണ്ടുമാണ് ഇത്.

സ്വയം പരിശോധിക്കാം.

1. x = 5, y=3 ആയാൽ താഴെ കൊടുത്തിരിക്കുന്ന പ്രവർത്തനങ്ങളുടെ ഔട്ട്പുട്ട് പ്രവചിക്കുക
   a) x>=10<<y>=4, b) x>=1<<y>=3, c) x>=11y>=4, d)x>=1 11y>=3
2. p=5, q=3, x=2 ആയാൽ ചുവടെ ചേർക്കുന്ന പ്രയോഗങ്ങളുടെ ഔട്ട്പുട്ട് പ്രവചിക്കുക
   a) ++P-q×r/2 b) p×q--+r c) p-q-r×2+p d) p+=5×q+r×r/2

5.7. പ്രസ്താവനകൾ (Statements)

ഒരു ഭാഷയുടെ പഠനശ്രേണി എന്നത് അക്ഷരമാല, പദങ്ങൾ, ശൈലികൾ, വാക്യങ്ങൾ, ഖണ്ഡികകൾ തുടങ്ങിയവയാണ്. അതുപോലെ C++ന്റെ പഠനത്തിൽ അക്ഷരമാല (character set), ടോക്കണുകൾ (tokens), പദപ്രയോഗങ്ങൾ എന്നിവ നമ്മൾ മനസ്സിലാക്കി കഴിഞ്ഞു. പ്രസ്താവനകളുടെ സഹായത്തോടെ കമ്പ്യൂട്ടറുമായി യുക്തിപരമായും അർത്ഥവത്തായും സംവദിക്കാവുന്ന രീതിയിൽ നാമിപ്പോൾ എത്തിയിട്ടുണ്ട്. ഒരു പ്രോഗ്രാമിംഗ് ഭാഷയിലെ ഏറ്റവും ചെറിയ പ്രവർത്തന ഘടകമാണ് പ്രസ്താവനകൾ. ഒരു പ്രസ്താവന അവസാനിച്ചു എന്ന് സൂചിപ്പിക്കുവാൻ C++; (Semi column) ഉപയോഗിക്കുന്നു. C++ൽ വ്യത്യസ്ത ആവശ്യങ്ങൾക്കായി പ്രഖ്യാപന പ്രസ്താവനകൾ (declaration), വിലനൽകുന്ന (assignment) പ്രസ്താവനകൾ, ഇൻപുട്ട് (input) പ്രസ്താവനകൾ, നിയന്ത്രണ

പ്രസ്താവനകൾ (control), ഔട്ട്പുട്ട് (output) പ്രസ്താവനകൾ തുടങ്ങിയവ ഉപയോഗിക്കുന്നു. ഒരു C++പ്രോഗ്രാമിലെ ഓരോ പ്രസ്താവനക്കും അതിന്റേതായ ലക്ഷ്യങ്ങളുണ്ട്. ഇവയിൽ പ്രഖ്യാപന പ്രസ്താവനകൾ ഒഴികെയുള്ളവ ചില പ്രത്യേക പ്രവർത്തനങ്ങൾ കമ്പ്യൂട്ടർ ഉപയോഗിച്ച് ചെയ്യാനുള്ളവയാണ്. നിർവഹണ പ്രസ്താവനകൾ (execulable statements) കമ്പ്യൂട്ടറനുള്ള നിർദ്ദേശങ്ങളാണ്. നിയന്ത്രണപ്രസ്താവനകളുടെ പ്രവർത്തനം അധ്യായം 7ൽ ചർച്ച ചെയ്യാം.

മറ്റു ചില പ്രസ്താവനകളെ നമുക്കിവിടെ ചർച്ചചെയ്യാം.

5.7.1. പ്രഖ്യാപന പ്രസ്താവനകൾ (Declaration statement)

എല്ലാ ഉപയോക്തൃ നിർവചിത വാക്കുകളും പ്രോഗ്രാമിൽ അവ ഉപയോഗിക്കുന്നതിനു മുൻപുതന്നെ നിർവചിക്കേണ്ടതാണ്. ഒരു വേരിയബിൾ എന്നത് ഉപയോക്താവ് നിർവചിക്കുന്നതാണെന്നും മെമ്മറിയിലെ ഒരിടത്തിനെ സൂചിപ്പിക്കുന്നതാണെന്നും നാം കണ്ടു. ഉപയോഗിക്കുന്നതിന് മുൻപ് പ്രോഗ്രാമിൽ ഇവ പ്രഖ്യാപിക്കപ്പെടേണ്ടതുണ്ട്. നാം ഒരു വേരിയബിളിനെ പ്രഖ്യാപിക്കുമ്പോൾ അതിൽ സൂക്ഷിച്ചിരിക്കുന്ന ഡാറ്റയുടെ ഇനം ഏതാണെന്ന് കംപൈലറിനെ അറിയിക്കുകയാണ് ചെയ്യുന്നത്. വേരിയബിൾ പ്രഖ്യാപിക്കുന്നതിന്റെ വാക്യഘടന:

Data Type<variable>, [<variable 2>, < variable 3>...];

Data Type എന്നത് C++ലെ ഏതെങ്കിലും അംഗീകൃതമായ ഡാറ്റ ഇനം ആകാം. ഒന്നിലധികം വേരിയബിളുകൾ പ്രയോഗിക്കുമ്പോൾ അവയെ വേർതിരിക്കാൻ കോമ (,) ഉപയോഗിക്കണം. ഒരു പ്രഖ്യാപന പ്രസ്താവന അർദ്ധവിരാമം (;) ത്തോട് കൂടി അവസാനിക്കുന്നു. സാധാരണയായി വേരിയബിളുകൾ പ്രഖ്യാപിക്കുന്നത് അവ ഉപയോഗിക്കുന്നതിന് തൊട്ട് മുൻപോ അല്ലെങ്കിൽ പ്രോഗ്രാമിന്റെ തുടക്കത്തിലോ ആയിരിക്കും. വാക്യഘടനയിൽ [ ] ൽ നൽകിയിരിക്കുന്നത് ആവശ്യമുണ്ടെങ്കിൽ മാത്രം ഉപയോഗിച്ചാൽ മതി എന്ന അർത്ഥത്തിലാണ്. താഴെ കൊടുത്തിരിക്കുന്ന പ്രസ്താവനകൾ വേരിയബിൾ പ്രഖ്യാപനങ്ങൾക്ക് ഉദാഹരണങ്ങളാണ്:

```
int roll number;

double wgpa, avg-score;
```

ഒന്നാമത്തെ പ്രസ്താവനയിൽ വേരിയബിൾ roll number ഒരു int ഡാറ്റ ഇനമായതിനാൽ ഇതിനായി 4 ബൈറ്റ് മെമ്മറി മാറ്റിവക്കപ്പെടുന്നു. (gcc അനുസരിച്ച്) ഇതിൽ 2147483648 മുതൽ +2147483647 വരെയുള്ള ഏതെങ്കിലും പൂർണ്ണസംഖ്യ സൂക്ഷിക്കാവുന്നതാണ്.

രണ്ടാമത്തെ പ്രസ്താവന wgpa, avg-score എന്നീ double ഡാറ്റ ഇനത്തിലുള്ള വേരിയബിളുകൾ നിർവചിക്കുന്നു. ഇവ ഓരോന്നിനും 8 ബൈറ്റ് മെമ്മറി വീതം നീക്കി വയ്ക്കുന്നു. പ്രോഗ്രാം കംപൈൽ ചെയ്യുന്ന സമയത്ത് ഇവക്കുള്ള മെമ്മറി നീക്കി വയ്ക്കുന്നു.

5.7.2 അസൈൻമെന്റ് പ്രസ്താവനകൾ (Assignment statement)

ഒരു വേരിയബിളിലേക്ക് വില നൽകുന്നതിനാണ് അസൈൻമെന്റ് ഓപ്പറേറ്റർ (=) ഉപയോഗിക്കുന്നത്. ഇങ്ങനെ ഉപയോഗിക്കുന്നതിനുള്ള പ്രസ്താവനകളെ അസൈൻമെന്റ് പ്രസ്താവന എന്ന് വിളിക്കുന്നു. താഴെപറയുന്ന ഏതെങ്കിലും രീതികളിൽ അവ എഴുതാം.

```
variable = constant;
variable1 = variable2;
variable = expression;
```

ഒന്നാമത്തേതിൽ ഒരു സ്ഥിരാംഗം വേരിയബിളിൽ സംഭരിക്കുന്നു. രണ്ടാമത്തേതിൽ വേരിയബിളിന്റെ വില മറ്റൊരു വേരിയബിളിൽ സംഭരിക്കുന്നു. മൂന്നാമത്തേതിൽ പദപ്രയോഗത്തിന്റെ ഫലം വേരിയബിളിൽ സംഭരിക്കുന്നു. അതുപോലെ നാലാമത്തേതിൽ ഫങ്ഷൻ തിരിച്ചുനൽകുന്ന വിലയാണ് വേരിയബിളിലേക്ക് സംഭരിക്കുന്നത്. ഫങ്ഷൻ എന്ന ആശയത്തെക്കുറിച്ച് അധ്യായം 10ൽ ചർച്ച ചെയ്യാം.

അസൈൻമെന്റ് പ്രസ്താവനകൾക്കുള്ള ചില ഉദാഹരണങ്ങൾ താഴെ കൊടുത്തിരിക്കുന്നു.

```
A = 15;   b = 5.8;
c = a+b;  d = (a+b) & (c+d)
r = sqrt (25);
```

അവസാനം നൽകിയിരിക്കുന്ന ഉദാഹരണത്തിൽ sqrt ( ) എന്നത് ഒരു ഫങ്ഷനാണ്. r എന്ന വേരിയബിളിൽ 25 ന്റെ വർഗമൂലമായ 5 ആണ് സംഭരിക്കപ്പെടുക. അസൈൻമെന്റ് പ്രസ്താവനകളിൽ ഇടതുവശത്ത് ഒരു വേരിയബിൾ തന്നെ ആയിരിക്കണം. പ്രോഗ്രാം പ്രവർത്തിക്കുമ്പോൾ ആദ്യം വലതുവശം പ്രവർത്തിച്ചശേഷം കിട്ടുന്ന ഫലം ഇടതുവശത്തെ വേരിയബിളിൽ (RHS) സംഭരിക്കുന്നു.

താഴെകാണിച്ചിരിക്കുന്നതുപോലെ ഒന്നിൽ കൂടുതൽ അസൈൻമെന്റുകൾ കൂട്ടിച്ചേർത്ത് ഒരേ സമയം ചെയ്യാവുന്നതാണ്. ഉദാഹരണത്തിന് x = y = z =13; ഇവിടെ 13 എന്ന വില z,y,x എന്നീ ക്രമത്തിൽ മൂന്ന് വേരിയബിളുകൾക്കും നൽകുന്നു. അസൈൻമെന്റ് പ്രസ്താവനയ്ക്കുമുമ്പ് വേരിയബിളുകൾ പ്രഖ്യാപിച്ചിരിക്കണം. ഒരു വേരിയബിളിന് നാം വില നൽകുകയാണെങ്കിൽ അതിലുള്ള പഴയ വില മാറ്റി പുതിയ വില നൽകുന്നു.

5.7.3 ഇൻപുട്ട് പ്രസ്താവന (Input statement):-

പ്രോഗ്രാമിന്റെ പ്രവർത്തനസമയത്ത് ഉപയോക്താവിന് ഡാറ്റ മെമ്മറിയിൽ സംഭരിക്കുന്നതിന് ഉപയോഗിക്കുന്ന പ്രസ്താവനകൾ ഇൻപുട്ട് പ്രസ്താവനകൾ എന്നറിയപ്പെടുന്നു. ഗെറ്റ് ഫ്രം, എക്സ്ട്രാക്ഷൻ എന്നീ പേരുകളിലറിയപ്പെടുന്ന >> ഓപ്പറേറ്ററാണ് ഇൻപുട്ട് ഓപ്പറേറ്റർ എന്നും നാം കണ്ടതാണ്. ഡാറ്റ സൂക്ഷിക്കേണ്ട RAMലെ സ്ഥാനവും ഇൻപുട്ട് നൽകുന്ന ഉപകരണവുമാണ് ഇൻപുട്ട് ഓപ്പറേറ്ററുടെ രണ്ട് ഓപെറന്റുകൾ. ഒരു അംഗീകൃത ഇൻപുട്ട് ഉപകരണമായ കീബോർഡിൽ നിന്ന് വരുന്ന തുടർച്ചയായ ഡാറ്റാ പ്രവാഹത്തെ വേരിയബിളുകൾ സൂചിപ്പിക്കുന്ന മെമ്മറി സ്ഥാനങ്ങളിൽ സംഭരിക്കപ്പെടുന്നു.. c++ ഒരു ഒബ്ജക്റ്റ് ഓറിയന്റഡ് ഭാഷയായതിനാൽ കീബോർഡ് ഒരു അംഗീകൃത ഇൻപുട്ട് സ്ട്രീം ഉപകരണമായാണ് കണക്കാക്കപ്പെടുന്നത്. സി ഇൻ (cin) എന്ന പേരിലുള്ള ഒരു ഒബ്ജക്റ്റ് ആയി തിരിച്ചറിയപ്പെടുകയും ചെയ്യുന്നു. ഒരു ഇൻപുട്ട് പ്രസ്താവനയുടെ ലളിതമായ രൂപം താഴെ കൊടുത്തിരിക്കുന്നു.

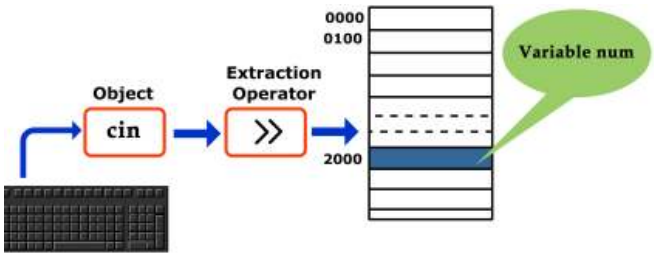
```
streamobject >> variable;
```

കീ ബോർഡ് ഒരു ഇൻപുട്ട് ഉപകരണമായി നാം ഉപയോഗിക്കുന്നതിനാൽ മുകളിൽ പറഞ്ഞ വാക്യഘടനയിൽ stream object നു പകരം cin എന്നു എഴുതുന്നു. >> എന്ന ഓപ്പ

റേറ്ററിനു നിർബന്ധമായും ഒരു വേരിയബിൾ തന്നെയാകണം ഓപ്പറന്റ് ആയി ഉപയോഗിക്കേണ്ടത്.ഉദാഹരണത്തിന് താഴെ പറയുന്ന പ്രസ്താവന കീബോർഡിൽ നിന്ന് ഡാറ്റാ സ്വീകരിക്കുകയും Num എന്ന വേരിയബിളിൽ സൂക്ഷിക്കുകയും ചെയ്യുന്നു.

cin>>num;

ഡാറ്റ കീ ബോർഡിൽ നിന്നും സ്വീകരിച്ച് എങ്ങനെ വേരിയബിളിൽ സംഭരിക്കുന്നു എന്ന് ചിത്രം 5.3. ൽ കാണിച്ചിരിക്കുന്നു.



ചിത്രം 5.3 C++

5.7.4 ഔട്ട്പുട്ട് പ്രസ്താവന (Output statement):-

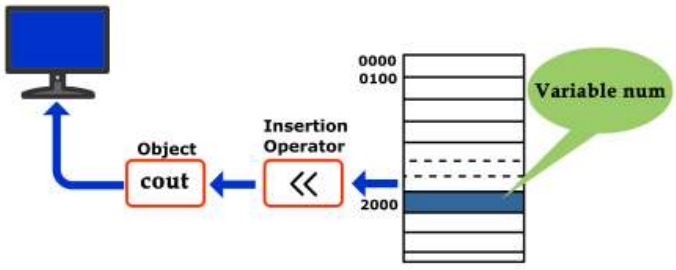
ഏതൊരു ഔട്ട്പുട്ട് ഉപകരണത്തിലൂടെയും ഉപയോക്താക്കൾക്ക് ഫലം ലഭ്യമാക്കുന്നതാണ് ഔട്ട്പുട്ട് പ്രസ്താവന. പുട്ട് ടു അല്ലെങ്കിൽ ഇൻസേർഷൻ എന്നീ പേരുകളിൽ അറിയപ്പെടുന്ന ഓപ്പറേറ്ററാണ് ഈ പ്രവർത്തനത്തിന് ഉപയോഗിക്കുന്നത്. ഇവിടെ ഔട്ട്പുട്ട് ചെയ്യേണ്ട ഡാറ്റയും ഔട്ട്പുട്ട് ഉപകരണവുമാണ് രണ്ട് ഓപെറന്റുകൾ. ഔട്ട്പുട്ട് പ്രസ്താവനയുടെ വാക്യഘടന ഇതാണ്.

streamobject << data;

stream object ഏതെങ്കിലും ഔട്ട്പുട്ട് ഉപകരണമാകാം. Data ഒരു സ്ഥിരാംഗമോ ഒരു വേരിയബിളോ അല്ലെങ്കിൽ ഒരു പദപ്രയോഗമോ ആകാം. മോണിറ്റർ ആണ് സാധാരണ ഉപയോഗിക്കുന്ന ഔട്ട്പുട്ട് ഉപകരണം. C++ ൽ cout (സി ഔട്ട് എന്ന് ഉച്ചരിക്കുന്നു) എന്നതാണ് മോണിറ്ററിനെ സൂചിപ്പിക്കുന്ന ഒബ്ജക്റ്റിന്റെ പേര്. മോണിറ്റർ ഔട്ട്പുട്ട് ഉപകരണമായി ഉപയോഗിക്കുന്ന ചില ഔട്ട്പുട്ട് പ്രസ്താവനകൾക്കുള്ള ഉദാഹരണങ്ങളാണ് താഴെ പറയുന്നവ.

```
cout << num;
cout << "hello friends";
cout << num+12;
```

ഒന്നാമത്ത പ്രസ്താവന num ന്റെ വില മോണിറ്ററിൽ പ്രദർശിപ്പിക്കുന്നു. രണ്ടാമത്തേത് hello friends എന്ന സ്ട്രിംഗ് പ്രദർശിപ്പിക്കുന്നു. അവസാനത്തേതിൽ num നോടുകൂടി 12 കൂട്ടി കിട്ടുന്ന ഫലം പ്രദർശിപ്പിക്കുന്നു. (num ൽ സംഖ്യയാണെന്ന് കരുതുക). മെമ്മറി സ്ഥാനം num ൽ നിന്ന് ഡാറ്റ എങ്ങനെയാണ് stream object (മോണിറ്റർ)ൽ ചേർത്തിരിക്കുന്നത് എന്ന് ചിത്രം 5.4. ൽ കാണിച്ചിരിക്കുന്നു.



ചിത്രം 5.4: C++

ടോക്കണുകളായ cin ഉം cout ഉം കിവേഡുകളല്ല. C++ ഭാഷയുടെ ഭാഗമല്ലാത്ത മുൻ നിർവചിത വാക്കുകളാണിവ. ഉപയോക്താവിന് ഇവയെ പുനർ വ്യാഖ്യാനം ചെയ്യാനവുന്നതാണ്. C++ ഭാഷയുടെ ലൈബ്രറിയിൽ നിർവ്വചിച്ചിട്ടുള്ള അടിസ്ഥാനവാക്കണിത്. വ്യക്തമായി പറയുകയാണെങ്കിൽ മുൻകൂട്ടി വായിച്ച അർത്ഥമുള്ള വാക്കുകളെ പുർവ്യാഖ്യാനം ചെയ്ത് മാറ്റാവശ്യങ്ങൾക്ക് ഉപയോഗിക്കുന്നത് അപകടകരവും ചിന്താകുഴപ്പം വരുത്തുന്നതുമാണ്. ഇത് ഒഴിവാക്കേണ്ടതാണ്. ഏറ്റവും ലളിതവും സുരക്ഷിതവുമായ മാർഗ്ഗം എന്നത് എല്ലാ മുൻ നിർവചിത ഐഡന്റിഫയറുകളെയും കൈകാര്യം ചെയ്യുന്നതാണ്.

കാസ്കേഡിംങ്ങ് ഓഫ് ഇൻപുട്ട് / ഔട്ട്പുട്ട് ഓപ്പറേറ്ററുകളുടെ (Cascading of I/O operators)

ഒന്നിൽകൂടുതൽ ഇൻപുട്ട് ഔട്ട്പുട്ട് ഓപ്പറേറ്റുകൾ ഒരുമിച്ച് ഉപയോഗിക്കുന്ന വിധം.

നമുക്ക് x,y,z എന്നീ മൂന്നു പേരുകളിലായി ഡാറ്റ ഇൻപുട്ട് ചെയ്യുന്നതിനായി

```
cin>>x;
cin>>y;
cin>>z;
```

ഇങ്ങനെ 3 പ്രസ്താവനകൾ ഉപയോഗിക്കാം; താഴെ പറയുന്ന രീതിയിൽ ഇവ മൂന്നും കൂട്ടി യോജിപ്പിച്ച് ഒറ്റ പ്രസ്താവനയായി ഉപയോഗിക്കാവുന്നതാണ്.

```
cin>>x>>y>>z;
```

ഒന്നിൽകൂടുതൽ ഇൻപുട്ട് ഔട്ട്പുട്ട് ഓപ്പറേറ്ററുകൾ ഒറ്റ പ്രസ്താവനയിൽ ഉപയോഗിക്കുന്നതിന് സംയോജിപ്പിച്ച് ഉണ്ടാക്കുമ്പോൾ ഓഫ് ഇൻപുട്ട് ഔട്ട്പുട്ട് ഓപ്പറേറ്ററുകളുടെ സംയോജനം എന്നു പറയുന്നു. ഇൻപുട്ട് ഓപ്പറേറ്ററുകൾ ചെയ്യുമ്പോൾ ആദ്യം നൽകുന്ന വില ആദ്യത്തെ വേരിയബിളിന് ലഭിക്കും. രണ്ടാമത്തേ വില രണ്ടാമത്തേതിന് അങ്ങനെ ഇടത്തുനിന്ന് വലത്തേക്ക് വില ലഭിക്കും. ഉദാഹരണമായി `cin>>x>>y>>z;` ഒന്നാമത് നൽകുന്ന വില x നും രണ്ടാമത്തേത് y ക്കും മൂന്നാമത്തേത് z നും ലഭിക്കും. പ്രവർത്തനസമയത്ത് വിലനൽകുമ്പോൾ വേരിയബിളുകളുടെ വിലകൾ തമ്മിൽ വേർതിരിക്കുന്നതിന് സ്പൈസ്, ബാർ, ടാബ്, അല്ലെങ്കിൽ എന്റർ കീ ഇവ ഏതെങ്കിലും ഉപയോഗിക്കാം.

ഇതുപോലെ ഒന്നിലധികം വേരിയബിളുകളുടെ വിലകൾ മോണിറ്ററിൽ കാണിക്കുന്നതിനായി താഴെ പറയുന്ന രീതി ഉപയോഗിക്കാം.

```
cout<<x<<y<<z;
```

വേരിയബിളുകൾ സ്ഥിരാംഗങ്ങൾ പദപ്രയോഗങ്ങൾ എന്നിവ ഒരുമിച്ച് ഔട്ട്പുട്ട് ചെയ്യാനായി താഴെ പറയുന്ന രീതി ഉപയോഗിക്കാം.

```
cout<<"The number is "<<z;
```

ഔട്ട്പുട്ട് ഓപ്പറേറ്ററുകൾ കാസ്കേഡ് ചെയ്യുമ്പോൾ വലത്തുനിന്ന് ഇടത്തേക്കായിരിക്കും ഔട്ട്പുട്ട് വിലകൾ പ്രദർശിപ്പിക്കുന്നത്.

'<<', '>>' എന്നീ ഓപ്പറേറ്ററുകളെ ഒരേ പ്രസ്താവനയിൽ ഉപയോഗിക്കാൻ കഴിയില്ല എന്നത് ശ്രദ്ധിക്കുമല്ലോ.

നമുക്ക് സംഗ്രഹിക്കാം

ഡാറ്റയുടെ തരം തിരിച്ചറിയുന്നതിനും അവയുമായി ബന്ധപ്പെട്ട ക്രിയകളെ കൈകാര്യം ചെയ്യുന്നതിനുമുള്ള ഒരു ഉപാധിയാണ് ഡാറ്റ ഇനങ്ങൾ. ഓരോ ഡാറ്റ ഇനത്തിലെ ഡാറ്റയ്ക്കും അതിന്റേതായ വലുപ്പവും പരിധിയുമുണ്ട്. വേരിയബിളുകൾ നിർവചിക്കാൻ ഡാറ്റ ഇനങ്ങൾ ഉപയോഗിക്കുന്നു. C++ ൽ വിവിധ ക്രിയകൾക്കായി വ്യത്യസ്തതരം ഓപ്പറേറ്ററുകൾ ലഭ്യമാണ്. ഓപ്പറേറ്ററുകൾ ഓപ്പറന്റുകളുമായി (ഡാറ്റ) കൂട്ടി ചേർക്കുമ്പോൾ പ്രയോഗങ്ങൾ രൂപപ്പെടുന്നു. മൂന്നു തരത്തിലുള്ള പ്രയോഗങ്ങളാണുള്ളത് - അരിത്മാറ്റിക്, റിലേഷണൽ, ലോജിക്കൽ. ഒരു പ്രോഗ്രാമിന്റെ ഏറ്റവും ചെറിയ പ്രവർത്തന ഭാഗമാണ് പ്രസ്താവന. വേരിയബിളിനെ പ്രഖ്യാപിക്കുന്ന പ്രസ്താവനകൾ പ്രോഗ്രാമിൽ ഒരു വേരിയബിളിനെ നിർവചിക്കുകയും അവക്ക് മെമ്മറി സ്ഥാനം നീക്കി വയ്ക്കുകയും ചെയ്യുന്നു. വില നൽകൽ പ്രസ്താവനകൾ, ഇൻപുട്ട് പ്രസ്താവനകൾ, ഔട്ട്പുട്ട് പ്രസ്താവനകൾ മുതലായവ കമ്പ്യൂട്ടറുകൾക്ക് നിർദ്ദേശങ്ങൾ നൽകാൻ സഹായിക്കുന്നു.

പഠന നേട്ടങ്ങൾ

ഈ അധ്യായത്തിൽ പൂർത്തികരണത്തോടെ പഠിതാവിന്

- C++ ലെ വിവിധ ഡാറ്റ ഇനങ്ങൾ തിരിച്ചറിയാൻ സാധിക്കുന്നു.
- ഉചിതമായ വേരിയബിളുകൾ തിരഞ്ഞെടുക്കുവാൻ സാധിക്കുന്നു.
- വിവിധ ഓപ്പറേറ്ററുകൾ പരീക്ഷിച്ചു നോക്കുവാൻ സാധിക്കുന്നു.
- വിവിധ I/O ഓപ്പറേറ്ററുകൾ പ്രയോഗിക്കുവാൻ സാധിക്കുന്നു.
- വിവിധ പ്രയോഗങ്ങളും പ്രസ്താവനകളും എഴുതുവാൻ സാധിക്കുന്നു.

മാതൃക ചോദ്യങ്ങൾ

ഹ്രസ്വോത്തര ചോദ്യങ്ങൾ

1. ഡാറ്റ ഇനം എന്നാലെന്ത്? C++ ലെ മുൻ നിർവചിത ഡാറ്റ ഇനങ്ങളുടെ പേരെഴുതുക.
2. void ഡാറ്റ ഇനത്തിന്റെ ഉപയോഗമെന്ത്?
3. കോൺസ്റ്റന്റ് (constant) എന്നാലെന്ത്?
4. ഡയനാമിക് ഇനിഷ്യലൈസേഷൻ എന്നാലെന്ത്?
5. ഓപ്പറേറ്ററിന്റെ നിർവചനം എഴുതുക.
6. യൂനറി ഓപ്പറേഷൻ എന്നാലെന്ത്?

7. പ്രഖ്യാപന പ്രസ്താവന എന്നാലെന്ത്?
8. ">>" (ഇൻപുട്ട് ഓപ്പറേറ്റർ), "<<" (ഔട്ട്പുട്ട് ഓപ്പറേറ്റർ) എന്നിവയുടെ പേരെഴുതുക.
9. താഴെ കൊടുത്തിരിക്കുന്നവ പരിഗണിച്ച് a = 5/3 എന്ന പ്രയോഗത്തിന്റെ ഉത്തരം എഴുതുക.
   i. a ഒരു float ഡാറ്റ ഇനം ആണെങ്കിൽ
   ii. a ഒരു int ഡാറ്റ ഇനം ആണെങ്കിൽ
10. i = 4, j = 5, k = 2 ആണെങ്കിൽ താഴെ തന്നിരിക്കുന്ന പ്രയോഗങ്ങളുടെ ഉത്തരം എഴുതുക
    (i) (5*++j)%6 (ii) (5*j++)%6
11. താഴെ കൊടുത്തിരിക്കുന്ന പ്രയോഗങ്ങളിലെ ഓപ്പറേറ്ററുകളുടെ മുൻഗണനാക്രമം എഴുതുക.
    (i) i+5>=j-6 (ii) s+10<p-2+2*q
12. "ans" ന്റെ വില 6 ആണെങ്കിൽ താഴെ തന്നിരിക്കുന്ന പ്രയോഗങ്ങളുടെ ഉത്തരം എഴുതുക
    (i) cout <<ans = 8 ; (ii) cout << ans == 8

ലഘു ഉപന്യാസ ചോദ്യങ്ങൾ

1. വേരിയബിൾ എന്നാലെന്ത്? വേരിയബിളുമായി ബന്ധപ്പെട്ടിട്ടുള്ള രണ്ട് വിലകൾ ഏതെല്ലാം?
2. ലോജിക്കൽ ഓപ്പറേറ്ററുകൾ വിശദീകരിക്കുക.
3. താഴെ തന്നിട്ടുള്ള C++ പ്രസ്താവനകളിൽ തെറ്റുകൾ ഉണ്ടെങ്കിൽ കണ്ടെത്തി കാരണം വിശദമാക്കുക.

   (i) `cout<< "a=" a;`
   (ii) `m=5,n=12;015`
   (iii) `cout << "x" ; <<x;`
   (iv) `cin >> y`
   (v) `cin >> "\n" >> y ;`
   (vi) `cout >> \n "abc"`
   (vii) `a = b + c`
   (viii) `break = x`

4. റിലേഷണൽ ഓപ്പറേറ്ററുകളുടെ ധർമ്മമെന്ത്? "=", "==" എന്നിവ താരതമ്യം ചെയ്യുക.

ഉപന്യാസ ചോദ്യങ്ങൾ

1. C++ ൽ ഉപയോഗിക്കുന്ന ഓപ്പറേറ്ററുകൾ വിശദീകരിക്കുക.
2. C++ ലെ പ്രയോഗങ്ങൾ വിശദീകരിക്കുക.