ਸੀ ਭਾਸ਼ਾ ਨਾਲ ਜਾਣ ਪਛਾਣ ਅਤੇ ਇਸ ਦੇ ਪ੍ਰੋਗਰਾਮਾਂ ਦੀ ਮੁਢਲੀ ਬਣਤਰ

ਪਾਠ - 7

ਇਸ ਪਾਠ ਦੇ ਉਦੇਸ

7.1 ਸੀ ਭਾਸ਼ਾ ਨਾਲ ਜਾਣ - ਪਛਾਣ ਅਤੇ ਇਤਿਹਾਸ
7.2 ਸੀ ਭਾਸ਼ਾ ਨੂੰ ਮਿਡਲ ਲੇਵਲ ਭਾਸ਼ਾ ਕਿਉ ਕਿਹਾ ਜਾਂਦਾ ਹੈ ?
7.3 ਸੀ ਭਾਸ਼ਾ ਦੇ ਐਡੀਟਰ ਅਤੇ IDE ਨਾਲ ਜਾਣ ਪਛਾਣ
7.4 ਸੀ ਭਾਸ਼ਾ ਦਾ ਪ੍ਰੋਗਰਾਮ ਬਨਾਉਣਾ ਅਤੇ ਚਲਾਉਣਾ
7.5 ਸੀ ਨਾਲ ਸ਼ੁਰੂਆਤ ਕਰਨਾ
7.6 ਕਰੈਕਟਰ ਸੈੱਟ
7.7 ਟੋਕਨਜ਼: ਕੀਅਵਰਡ, ਆਈਡੈਂਟੀਫਾਇਰ, ਲਿਟਰਲਜ਼, ਆਪਰੇਟਰਜ਼, ਖਾਸ ਚਿੰਨ੍ਹ
7.8 ਵੇਰੀਏਬਲ ਅਤੇ ਕਾਂਸਟੈਂਟ ਦੀ ਧਾਰਣਾ ਅਤੇ ਇਹਨਾਂ ਦੀ ਡਿਕਲੇਰੇਸ਼ਨ
7.9 ਡਾਟਾ ਟਾਈਪਸ (ਕੇਵਲ ਪ੍ਰੀਮੀਟਿਵ ਡਾਟਾ ਟਾਈਪਸ)
7.10 ਸੀ ਵਿੱਚ ਹੈਡਰ ਫਾਈਲਾਂ
7.11 ਸੀ ਵਿੱਚ ਇਨਪੁੱਟ ਅਤੇ ਆਉਟਪੁੱਟ ਸਟੇਟਮੈਂਟਸ
7.12 ਸੀ ਭਾਸ਼ਾ ਦੇ ਪ੍ਰੋਗਰਾਮ ਦੀ ਬਣਤਰ

7.1 ਸੀ ਭਾਸ਼ਾ ਨਾਲ ਜਾਣ ਪਛਾਣ ਅਤੇ ਇਤਿਹਾਸ (Introduction and History of C) :

ਸੀ ਇੱਕ **ਜਨਰਲ-ਪਰਪਜ਼ ਪ੍ਰੋਗਰਾਮਿੰਗ** ਭਾਸ਼ਾ ਹੈ। ਸੀ ਭਾਸ਼ਾ ਦੀ ਵਰਤੋਂ ਕਿਸੇ ਵੀ ਤਰਾਂ ਦੇ ਐਪਲੀਕੇਸਨ ਸਾਫਟਵੇਅਰ ਤਿਆਰ ਕਰਨ ਲਈ ਕੀਤੀ ਜਾ ਸਕਦੀ ਹੈ। ਇਸ ਵਿੱਚ ਅਸੀਂ ਬਿਜਨਸ ਐਪਲੀਕੇਸ਼ਨਾਂ, ਵਿਗਿਆਨਕ ਐਪਲੀਕੇਸ਼ਨਾਂ ਆਦਿ ਤਿਆਰ ਕਰ ਸਕਦੇ ਹਾਂ। ਇਸਦੀ ਵਰਤੋਂ ਸਿਸਟਮ ਪ੍ਰੋਗਰਾਮ ਜਿਵੇਂ ਕਿ ਓਪਰੇਟਿੰਗ ਸਿਸਟਮ, ਭਾਸ਼ਾ ਟ੍ਰਾਂਸਲੇਟਰਾਂ ਆਦਿ ਨੂੰ ਤਿਆਰ ਕਰਨ ਲਈ ਵੀ ਕੀਤੀ ਜਾ ਸਕਦੀ ਹੈ। ਇਸ ਤਰ੍ਹਾਂ ਅਸੀਂ ਕਹਿ ਸਕਦੇ ਹਾਂ ਕਿ ਇਹ ਦੋਵੇਂ ਤਰ੍ਹਾਂ ਦੇ ਸਾਫਟਵੇਅਰ, ਭਾਵ ਸਿਸਟਮ ਸਾਫਟਵੇਅਰ ਅਤੇ ਐਪਲੀਕੇਸਨ ਸਾਫਟਵੇਅਰ ਤਿਆਰ ਕਰਨ ਲਈ ਲਾਭਦਾਇੱਕ ਹੈ।

1960 ਵਿੱਚ ਬਹੁਤ ਸਾਰੀਆਂ ਕੰਪਿਊਟਰ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸਾਵਾਂ FORTRAN, COBOL ਆਦਿ ਉੱਭਰ ਕੇ ਸਾਹਮਣੇ ਆਈਆਂ ਸਨ। ਪਰ ਇਹ ਭਾਸਾਵਾਂ ਸਿਰਫ ਖਾਸ ਕਾਰਜਾਂ ਲਈ ਵਰਤੀਆਂ ਜਾਂਦੀਆਂ ਸਨ। ਉਦਾਹਰਣ ਦੇ ਲਈ: FORTRAN (Formula Translation) ਸਿਰਫ ਵਿਗਿਆਨਕ ਕਾਰਜਾਂ ਦੇ ਪ੍ਰੋਗਰਾਮਾਂ ਨੂੰ ਵਿਕਸਤ ਕਰਨ ਲਈ ਇਸਤੇਮਾਲ ਕੀਤੀ ਜਾਂਦੀ ਸੀ ਜਦੋਂ ਕਿ COBOL (Common Business Oriented Language) ਸਿਰਫ ਵਪਾਰਕ ਕਾਰਜਾਂ ਵਾਲੇ ਪ੍ਰੋਗਰਾਮਾਂ ਨੂੰ ਵਿਕਸਤ ਕਰਨ ਲਈ ਇਸਤੇਮਾਲ ਕੀਤੀ ਜਾਂਦੀ ਸੀ। ਬਾਅਦ ਵਿੱਚ ਆਮ ਉਦੇਸ਼ਾਂ ਦੀ ਪੂਰਤੀ ਕਰਨ ਵਾਲੀ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾ (General purpose programming language) ਨੂੰ ਵਿਕਸਤ ਕਰਨ ਲਈ ਇੱਕ ਅੰਤਰਰਾਸ਼ਟਰੀ ਕਮੇਟੀ ਬਣਾਈ ਗਈ ਸੀ। ਨਤੀਜੇ ਵਜੋਂ, 1963 ਵਿੱਚ ਕੈਂਬ੍ਰਿਜ ਯੂਨੀਵਰਸਿਟੀ (Cambridge University) ਵਿੱਚ ਸੰਯੁਕਤ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾ Combined Programming Language (CPL) ਤਿਆਰ ਕੀਤੀ ਗਈ। ਇਸ ਭਾਸ਼ਾ ਨੂੰ ਸਿੱਖਣਾ ਅਤੇ ਲਾਗੂ ਕਰਨਾ ਬਹੁਤ ਮੁਸਕਿਲ ਸੀ। ਇਸ ਲਈ ਬਾਅਦ ਵਿੱਚ, 1967 ਵਿੱਚ,

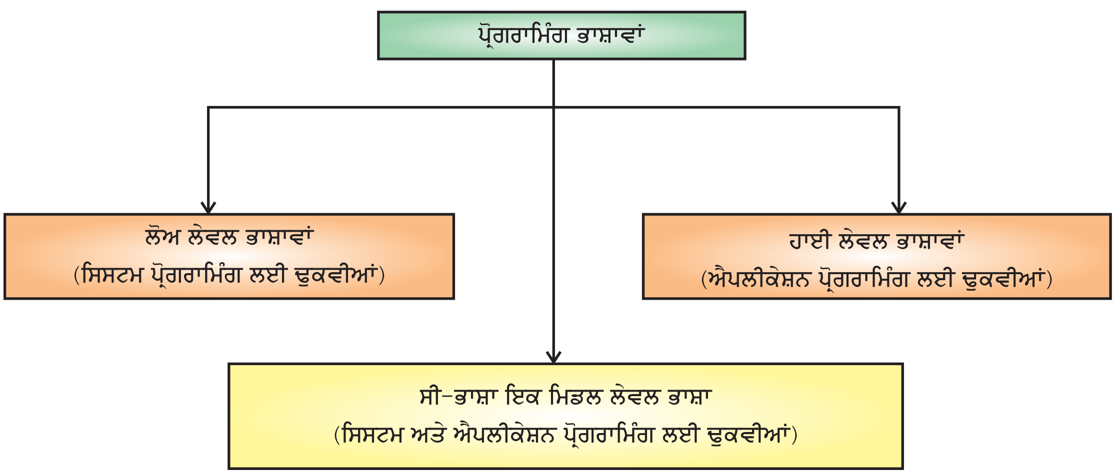
BCPL (Basic Combined Programming Language) ਭਾਸ਼ਾ ਮਾਰਟਿਨ ਰਿਚਰਡਜ ਦੁਆਰਾ ਕੈਂਬਰਿਜ ਯੂਨੀਵਰਸਿਟੀ ਵਿੱਚ ਤਿਆਰ ਕੀਤੀ ਗਈ ਸੀ। ਇਸੇ ਤਰ੍ਹਾਂ B ਭਾਸ਼ਾ ਨੂੰ ਕੇਨ ਥੌਮਸਨ ਨੇ 1970 ਵਿੱਚ AT & T ਬੈੱਲ ਲੈਬਾਰਟਰੀ ਵਿੱਚ ਵਿਕਸਿਤ ਕੀਤਾ ਸੀ। ਪਰ ਇਹ ਦੋਵੇਂ ਭਾਸਾਵਾਂ, BCPL ਅਤੇ B, ਟਾਈਪ-ਲੈੱਸ (type-less) ਭਾਸਾਵਾਂ ਸਨ ਅਤੇ ਬਹੁਤ ਹੀ ਵਿਸ਼ੇਸ਼ ਸਨ। ਅੰਤ ਵਿੱਚ, 1972 ਵਿੱਚ BCPL ਅਤੇ B ਭਾਸਾਵਾਂ ਦੀਆਂ ਮਹੱਤਵਪੂਰਣ ਵਿਸ਼ੇਸ਼ਤਾਵਾਂ ਦੀ ਵਰਤੋਂ ਕਰਦੇ ਹੋਏ ਡੈਨਿਸ ਰਿਚੀ ਦੁਆਰਾ ਸੰਯੁਕਤ ਰਾਜ ਅਮਰੀਕਾ ਦੀ AT & T ਬੈੱਲ ਲੈਬਾਰੋਟਰੀ ਵਿੱਚ C ਭਾਸਾ ਨੂੰ ਵਿਕਸਿਤ ਕੀਤਾ ਗਿਆ ਅਤੇ ਇਹ ਇੱਕ ਆਮ ਉਦੇਸ਼ ਵਾਲੀ (general-purpose) ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾ ਬਣ ਗਈ।

7.2 ਸੀ ਭਾਸ਼ਾ ਨੂੰ ਮਿਡਲ ਲੇਵਲ ਭਾਸ਼ਾ ਕਿਉਂ ਕਿਹਾ ਜਾਂਦਾ ਹੈ ? (Why C Language is called Middle Level Language) ?

ਸਾਰੀਆਂ ਹੀ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸਾਵਾਂ ਨੂੰ ਦੋ ਕਿਸਮਾਂ ਵਿੱਚ ਵੰਡਿਆ ਜਾ ਸਕਦਾ ਹੈ: ਲੋਅ-ਲੇਵਲ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾਵਾਂ ਅਤੇ ਹਾਈ ਲੇਵਲ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾਵਾਂ।

ਲੋਅ-ਲੇਵਲ ਭਾਸ਼ਾਵਾਂ ਨੂੰ ਮਸ਼ੀਨ-ਓਰੀਐਂਟਡ ਭਾਸ਼ਾਵਾਂ ਵਜੋਂ ਜਾਣਿਆ ਜਾਂਦਾ ਹੈ ਕਿਉਂਕਿ ਪ੍ਰੋਗਰਾਮਰ ਨੂੰ ਹੱਲ ਕੀਤੇ ਜਾਣ ਵਾਲੇ ਪ੍ਰੋਗਰਾਮ ਦੇ ਲਾਜਿਕ ਦੀ ਬਜਾਏ ਹਾਰਡਵੇਅਰ (ਮਸੀਨ) ਦੀ ਅੰਦਰੂਨੀ ਬਣਤਰ ਉੱਪਰ ਵਧੇਰੇ ਧਿਆਨ ਕੇਂਦਰਿਤ ਕਰਨਾ ਪੈਂਦਾ ਹੈ। ਇਹਨਾਂ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾਵਾਂ ਨੂੰ ਸਿੱਖਣਾ ਅਤੇ ਇਸਤੇਮਾਲ ਕਰਨਾ ਮੁਸ਼ਕਿਲ ਹੈ। ਇਹ ਭਾਸਾਵਾਂ ਮਸੀਨ ਉੱਪਰ ਨਿਰਭਰ (machine-dependent) ਸਿਸਟਮ-ਪ੍ਰੋਗਰਾਮਾਂ, ਜਿਵੇਂ ਕਿ ਓਪਰੇਟਿੰਗ ਸਿਸਟਮ, ਭਾਸ਼ਾ ਟ੍ਰਾਂਸਲੇਟਰ ਆਦਿ ਨੂੰ ਬਨਾਉਣ ਲਈ ਵਰਤੀਆਂ ਜਾਂਦੀਆਂ ਹਨ। ਇਨ੍ਹਾਂ ਭਾਸ਼ਾਵਾਂ ਦੀਆਂ ਉਦਾਹਰਣਾਂ ਅਸੈਂਬਲੀ ਭਾਸਾ ਅਤੇ ਮਸ਼ੀਨ ਭਾਸ਼ਾ ਹਨ।

ਹਾਈ ਲੇਵਲ ਭਾਸ਼ਾਵਾਂ ਨੂੰ ਸਮੱਸਿਆ-ਅਧਾਰਿਤ (Problem Oriented) ਭਾਸ਼ਾਵਾਂ ਵਜੋਂ ਜਾਣਿਆ ਜਾਂਦਾ ਹੈ ਕਿਉਂਕਿ ਪ੍ਰੋਗਰਾਮਰ ਨੂੰ ਮਸ਼ੀਨ ਦੀ ਅੰਦਰੂਨੀ ਬਣਤਰ ਜਾਨਣ ਦੀ ਬਜਾਏ ਸਮੱਸਿਆ ਨੂੰ ਹੱਲ ਕਰਨ ਦੇ ਲਾਜਿਕ ਤੇ ਵਧੇਰੇ ਧਿਆਨ ਕੇਂਦਰਿਤ ਕਰਨਾ ਪੈਂਦਾ ਹੈ। ਇਹਨਾਂ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾਵਾਂ ਨੂੰ ਸਿੱਖਣਾ ਅਤੇ ਇਸਤੇਮਾਲ ਕਰਨਾ ਆਸਾਨ ਹੁੰਦਾ ਹੈ। ਇਹ ਭਾਸ਼ਾਵਾਂ ਵੱਖ-ਵੱਖ ਕਿਸਮਾਂ ਦੇ ਐਪਲੀਕੇਸ਼ਨ ਪ੍ਰੋਗਰਾਮਾਂ ਨੂੰ ਲਿਖਣ ਲਈ ਵਰਤੀਆਂ ਜਾ ਸਕਦੀਆਂ ਹਨ, ਜਿਵੇਂ ਕਿ ਵਪਾਰਕ ਕਾਰਜਾਂ, ਵਿਗਿਆਨਕ ਕਾਰਜਾਂ ਆਦਿ ਨਾਲ ਸੰਬੰਧਤ ਪ੍ਰੋਗਰਾਮ ਬਨਾਉਣ ਲਈ। ਇਨ੍ਹਾਂ ਭਾਸ਼ਾਵਾਂ ਦੀਆਂ ਉਦਾਹਰਣਾਂ ਹਨ: FORTRAN, COBOL, Pascal, C++, Java ਆਦਿ।



ਚਿੱਤਰ 7.1 ਸੀ-ਭਾਸ਼ਾ - ਮਿਡਲ ਲੇਵਲ ਭਾਸ਼ਾ

ਸੀ ਭਾਸ਼ਾ ਵਿੱਚ ਇਹਨਾਂ ਦੋਵੇਂ ਕਿਸਮਾਂ ਦੀਆਂ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸਾਵਾਂ, ਭਾਵ ਲੋਅ-ਲੇਵਲ ਅਤੇ ਹਾਈ-ਲੇਵਲ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾਵਾਂ ਦੀ ਸਮਰੱਥਾ ਹੈ। ਇਸਦਾ ਅਰਥ ਇਹ ਹੈ ਕਿ ਸੀ ਭਾਸ਼ਾ ਦੋਵੇਂ ਤਰ੍ਹਾਂ ਦੇ ਪ੍ਰੋਗਰਾਮਾ- ਸਿਸਟਮ ਪ੍ਰੋਗਰਾਮ ਅਤੇ ਐਪਲੀਕੇਸ਼ਨ ਪ੍ਰੋਗਰਾਮਾਂ ਨੂੰ ਲਿਖਣ ਲਈ ਢੁੱਕਵੀਂ ਹੈ। ਇਸ ਤਰ੍ਹਾਂ ਸੀ-ਭਾਸ਼ਾ ਇੱਕ ਅਜਿਹੀ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾ ਹੈ ਜੋ ਉਪਰੋਕਤ ਦੋਵੇਂ ਕਿਸਮਾਂ ਦੀਆਂ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾਵਾਂ (ਹਾਈ ਲੇਵਲ ਅਤੇ ਲੋਅ ਲੇਵਲ) ਦੇ ਵਿੱਚਕਾਰ ਖੜ੍ਹੀ ਹੈ। ਇਸ ਲਈ ਸੀ ਭਾਸ਼ਾ ਨੂੰ **ਮਿਡਲ ਲੈਵਲ ਭਾਸ਼ਾ** ਕਿਹਾ ਜਾਂਦਾ ਹੈ।

ਹਾਲਾਂਕਿ, ਮਿਡਲ ਲੇਵਲ ਭਾਸ਼ਾ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾਵਾਂ ਦੀ ਕੋਈ ਵਿਸ਼ੇਸ਼ ਸ਼੍ਰੇਣੀ ਨਹੀਂ ਹੈ, ਇਹ ਸਿਰਫ ਸੀ ਭਾਸ਼ਾ ਦੀਆਂ ਵਿਸ਼ੇਸ਼ ਸਮਰੱਥਾਵਾਂ ਦੇ ਕਾਰਨ ਹੈ ਜੋ ਇਸਨੂੰ ਮਿਡਲ ਲੇਵਲ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾ ਵੱਜੋਂ ਜਾਣਿਆ ਜਾਂਦਾ ਹੈ।

7.3 ਸੀ ਭਾਸ਼ਾ ਦੇ ਐਡੀਟਰ ਅਤੇ IDEs ਨਾਲ ਜਾਣ ਪਛਾਣ (Intro to C Editors & IDEs)

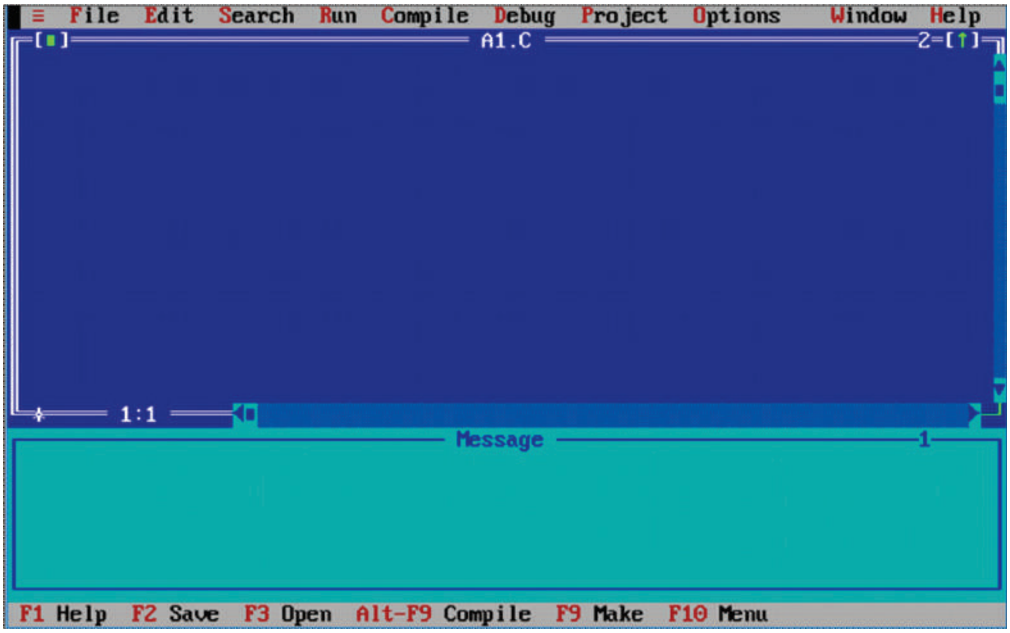
ਐਡੀਟਰ ਉਹ ਪ੍ਰੋਗਰਾਮ ਹੁੰਦੇ ਹਨ ਜੋ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸਾਵਾਂ ਦੇ ਸੋਰਸ ਕੋਡ ਲਿਖਣ ਲਈ ਲਾਭਦਾਇੱਕ ਹੁੰਦੇ ਹਨ। ਬਹੁਤ ਸਾਰੀਆਂ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾਵਾਂ ਅਜਿਹੀਆਂ ਹਨ ਜਿਨ੍ਹਾਂ ਦੇ ਪ੍ਰੋਗਰਾਮ ਲਿਖਣ ਲਈ ਉਹਨਾਂ ਦੇ ਆਪਣੇ ਐਡੀਟਰ ਹੁੰਦੇ ਹਨ, ਜਿਵੇਂ ਕਿ C, Pascal ਆਦਿ। ਪਰ ਕੁਝ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾਵਾਂ ਵਿੱਚ ਪ੍ਰੋਗਰਾਮ ਲਿਖਣ ਲਈ ਆਪਣਾ ਐਡੀਟਰ ਨਹੀਂ ਹੁੰਦਾ, ਜਿਵੇਂ ਕਿ Java ਆਦਿ। ਜਾਵਾ ਪ੍ਰੋਗਰਾਮ ਨੂੰ ਕਿਸੇ ਵੀ ਸਧਾਰਨ ਟੈਕਸਟ ਐਡੀਟਰ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਲਿਖਿਆ ਜਾ ਸਕਦਾ ਹੈ। ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾਵਾਂ ਵਿੱਚ ਸੋਰਸ-ਪ੍ਰੋਗਰਾਮ ਲਿਖਣ ਤੋਂ ਬਾਅਦ, ਉਹਨਾਂ ਨੂੰ ਕੰਪਿਊਟਰਾਂ ਦੁਆਰਾ ਚਲਾਉਣ ਯੋਗ ਬਨਾਉਣ ਲਈ ਕੰਪਾਇਲ ਕੀਤਾ ਜਾਂਦਾ ਹੈ। ਇਹਨਾਂ ਸੋਰਸ ਪ੍ਰੋਗਰਾਮਾਂ ਨੂੰ ਉਨ੍ਹਾਂ ਦੇ ਸੰਬੰਧਿਤ ਕੰਪਾਈਲਰਾਂ ਦੁਆਰਾ ਹੀ ਕੰਪਾਇਲ ਕੀਤਾ ਜਾ ਸਕਦਾ ਹੈ।

ਬਹੁਤ ਸਾਰੀਆਂ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸਾਵਾਂ **IDEs** ਦਾ ਪ੍ਰਯੋਗ ਵੀ ਕਰਦੀਆਂ ਹਨ। **IDE** ਦਾ ਅਰਥ **Integrated Development Environment (IDE)** ਹੈ। ਇੰਟੀਗਰੇਟਡ ਡਿਵੈਲਪਮੈਂਟ ਇਨਵਾਇਰਮੈਂਟ **(IDE)** ਨੂੰ ਇੱਕ ਅਜਿਹੇ ਸਾਫਟਵੇਅਰ ਵਜੋਂ ਪਰਿਭਾਸ਼ਤ ਕੀਤਾ ਜਾ ਸਕਦਾ ਹੈ ਜੋ ਇਸਦੇ ਯੂਜ਼ਰ ਨੂੰ ਪ੍ਰੋਗਰਾਮਾਂ ਨੂੰ ਲਿਖਣ ਦੇ ਵਾਤਾਵਰਣ (ਐਡੀਟਰ) ਦੇ ਨਾਲ-ਨਾਲ ਪ੍ਰੋਗਰਾਮਾਂ ਨੂੰ ਕੰਪਾਈਲ ਕਰਨ, ਚਲਾਉਣ, ਟੈਸਟ ਕਰਨ ਅਤੇ ਡੀਬੱਗ ਕਰਨ ਦੇ ਟੂਲਜ਼ ਵੀ ਪ੍ਰਦਾਨ ਕਰਦਾ ਹੈ। ਆਮ ਤੌਰ ਤੇ ਆਧੁਨਿਕ IDE ਸਾਫਟਵੇਅਰ ਬਹੁਤ ਉਪਭੋਗਤਾ-ਅਨੁਕੂਲ (User-Friendly) ਹੁੰਦੇ ਹਨ। ਇਹ ਵਰਤਣ ਵਿੱਚ ਅਸਾਨ-ਇੰਟਰਫੇਸ ਪ੍ਰਦਾਨ ਕਰਦੇ ਹਨ। ਇਹ IDE ਪ੍ਰੋਗਰਾਮਰ ਨੂੰ ਗ੍ਰਾਫਿਕਲ ਯੂਜਰ ਇੰਟਰਫੇਸ, ਸਿੰਟੈਕਸ ਸੰਬੰਧੀ ਸੁਝਾਅ, ਐਡੀਟਰ, ਪਲੱਗਇਨਸ (Plugins) ਅਤੇ ਹੋਰ ਬਹੁਤ ਸਾਰੀਆਂ ਵਿਸੇਸਤਾਵਾਂ ਵੀ ਪ੍ਰਦਾਨ ਕਰਦੇ ਹਨ।

C ਵਿੱਚ ਪ੍ਰੋਗਰਾਮਿੰਗ ਕਰਨ ਲਈ ਬਹੁਤ ਸਾਰੇ IDEs ਉਪਲੱਬਧ ਹਨ, ਜਿਨ੍ਹਾਂ ਵਿਚੋਂ ਕੁੱਝ ਆਮ ਵਰਤੇ ਜਾਣ ਵਾਲੇ IDEs ਹੇਠਾਂ ਦਿੱਤੇ ਗਏ ਹਨ:

- Turbo C
- Code Blocks
- Eclipse
- Code Lite
- Net Beans
- Dev C++ ਆਦਿ

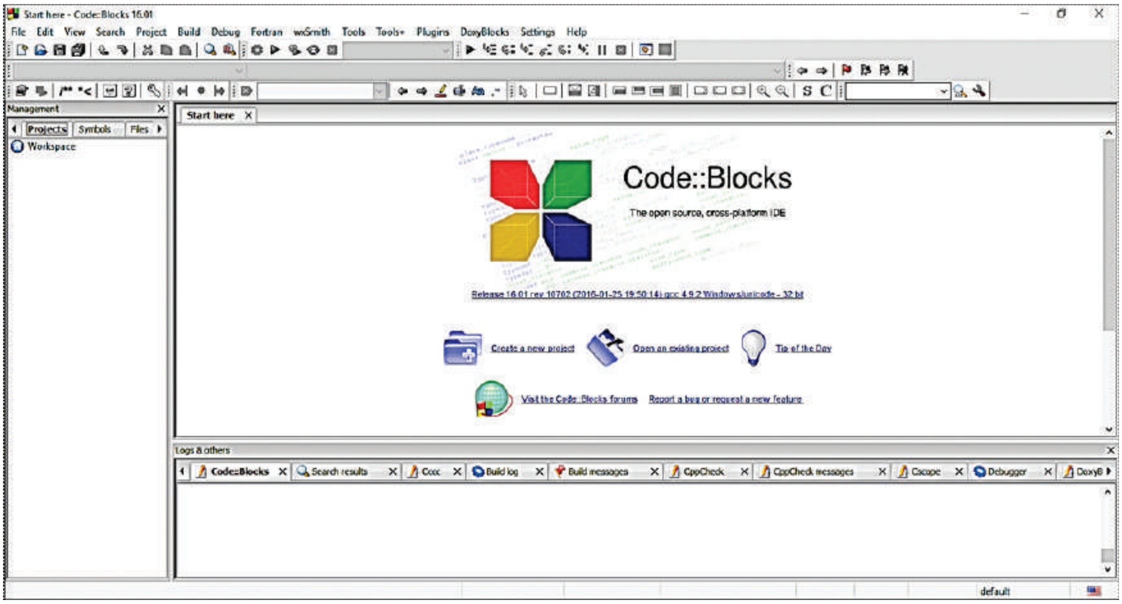
ਸੀ ਭਾਸ਼ਾ ਵਿੱਚ ਪ੍ਰੋਗਰਾਮਿੰਗ ਕਰਨ ਲਈ Turbo C ਇੱਕ ਬਹੁਤ ਪੁਰਾਣਾ IDE ਹੈ। ਇਹ ਬੋਰਲੈਂਡ (Borland) ਦੁਆਰਾ ਤਿਆਰ ਕੀਤਾ ਗਿਆ ਸੀ ਅਤੇ ਸਭ ਤੋਂ ਪਹਿਲਾਂ 1987 ਵਿੱਚ ਪੇਸ਼ ਕੀਤਾ ਗਿਆ ਸੀ। ਉਸ ਸਮੇਂ Turbo C



ਚਿੱਤਰ 7.2 Turbo C/C++ IDE ਇੰਟਰਫੇਸ

ਆਪਣੇ ਛੋਟੇ (Compact) ਅਕਾਰ, ਵਿਆਪਕ ਦਸਤਾਵੇਜ (Comprehensive manual), ਤੇਜ ਕੰਪਾਈਲਿੰਗ ਸਪੀਡ ਅਤੇ ਘੱਟ ਕੀਮਤ ਲਈ ਜਾਣਿਆ ਜਾਂਦਾ ਸੀ। ਪਰ Turbo C ਅੱਜ-ਕੱਲ ਦੇ ਨਵੇਂ ਓਪਰੇਟਿੰਗ ਸਿਸਟਮਾਂ ਜਿਵੇਂ ਕਿ ਵਿੰਡੋਜ 7, ਵਿੰਡੋਜ 8 ਅਤੇ ਵਿੰਡੋਜ 10 ਲਈ ਕਾਫੀ ਪੁਰਾਣਾ ਹੋ ਚੁਕਿਆ ਹੈ। ਇਹਨਾਂ ਓਪਰੇਟਿੰਗ ਸਿਸਟਮਾਂ ਵਿੱਚ ਅਨੁਕੂਲਤਾ ਦੇ ਮੁੱਦਿਆਂ (compatibility issues) ਕਾਰਨ ਪ੍ਰੋਗਰਾਮਰਾਂ ਨੂੰ Turbo C ਦੀ ਵਰਤੋਂ ਵਿੱਚ ਬਹੁਤ ਸਾਰੀਆਂ ਮੁਸ਼ਕਿਲਾਂ ਦਾ ਸਾਹਮਣਾ ਕਰਨਾ ਪੈ ਰਿਹਾ ਹੈ। ਸੀ ਵਿੱਚ ਪ੍ਰੋਗਰਾਮਿੰਗ ਲਈ ਬਹੁਤ ਸਾਰੇ ਆਧੁਨਿਕ IDE ਤਿਆਰ ਕੀਤੇ ਗਏ ਹਨ ਜੋ ਆਧੁਨਿਕ ਓਪਰੇਟਿੰਗ ਸਿਸਟਮਾਂ ਵਿੱਚ ਵਧੀਆ ਢੰਗ ਨਾਲ ਕੰਮ ਕਰਦੇ ਹਨ।

ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾ C/C++ ਵਿੱਚ ਪ੍ਰੋਗਰਾਮਿੰਗ ਲਈ Code :: Blocks ਇੱਕ ਪ੍ਰਸਿੱਧ ਅਤੇ ਆਧੁਨਿਕ IDE ਹੈ। ਇਹ ਇੱਕ ਮੁਫਤ ਓਪਨ ਸੋਰਸ, ਐਕਸਟੈਂਸੀਬਲ (extensible) ਅਤੇ ਕਨਫਿਗਰੇਬਲ (configurable), ਕਰਾਸ ਪਲੇਟਫਾਰਮ (cross plateform) IDE ਹੈ। ਇਹ ਪ੍ਰੋਗਰਾਮਰ ਨੂੰ ਆਦਰਸ਼ ਵਿਸ਼ੇਸ਼ਤਾਵਾਂ (ideal features) ਪ੍ਰਦਾਨ ਕਰਦਾ ਹੈ। ਇਹ ਇੱਕਸਾਰ (uniform) ਯੂਜ਼ਰ ਇੰਟਰਫੇਸ ਪ੍ਰਦਾਨ ਕਰਦਾ ਹੈ। ਸਭ ਤੋਂ ਵੱਧ ਮਹੱਤਵਪੂਰਨ ਗੱਲ ਇਹ ਹੈ ਕਿ ਅਸੀਂ ਹੋਰ ਯੂਜ਼ਰਾਂ ਦੁਆਰਾ ਤਿਆਰ ਕੀਤੇ ਪਲੱਗਇਨ (plugins) ਦੀ ਵਰਤੋਂ ਕਰਕੇ ਇਸ IDE ਦੀ ਕਾਰਜਸ਼ੀਲਤਾ (functionality) ਨੂੰ ਵਧਾ ਸਕਦੇ ਹਾਂ।



ਚਿੱਤਰ 7.3 Code::Blocks IDE ਦਾ ਇੰਟਰਫੇਸ

Code::Blocks ਪ੍ਰੋਗਰਾਮਰਾਂ ਨੂੰ ਬਹੁਤ ਸਾਰੀਆਂ ਸਹੂਲਤਾਂ ਪ੍ਰਦਾਨ ਕਰਦਾ ਹੈ ਜੋ ਹੇਠਾਂ ਦਿੱਤੀਆਂ ਗਈਆਂ ਹਨ:

- ਇਹ Windows, Linux ਅਤੇ Mac OS X ਉੱਪਰ ਵੀ ਕੰਮ ਕਰਦਾ ਹੈ।
- ਇਹ ਕੋਡ ਦੀ ਕੰਪਾਈਲੇਸ਼ਨ, ਡਿੱਬਗਿੰਗ, ਕਵਰੇਜ਼, ਪ੍ਰੋਫਾਈਲਿੰਗ, ਅਤੇ ਆਟੋ ਕੰਪਲੀਸ਼ਨ ਆਦਿ ਕੰਮਾਂ ਵਿੱਚ ਮਦਦ ਪ੍ਰਦਾਨ ਕਰਦਾ ਹੈ।
- ਇਹ ਕਈ ਕੰਪਾਈਲਰਾਂ ਨੂੰ ਸਪੋਰਟ (support) ਕਰਦਾ ਹੈ ਜਿਸ ਵਿੱਚ GCC, Mingw, clang, Borland C++ 5.5, ਆਦਿ ਸ਼ਾਮਲ ਹਨ।
- ਇਹ ਬਹੁਤ ਤੇਜ਼ ਹੈ, ਇਸ ਵਿੱਚ make files ਬਨਾਉਣ ਦੀ ਜਰੂਰਤ ਨਹੀਂ ਪੈਂਦੀ।
- ਇਹ ਡਿੱਬਗਿੰਗ ਲਈ ਬ੍ਰੇਕ-ਪੁਆਇੰਟਸ (breakpoints) ਨੂੰ ਪੂਰਨ ਰੂਪ ਵਿੱਚ ਸਪੋਰਟ (support) ਕਰਦਾ ਹੈ ਜਿਸ ਵਿੱਚ code breakpoints, data breakpoints, breakpoint conditions ਆਦਿ ਹੋਰ ਬਹੁਤ ਸਾਰੀਆਂ ਸੁਵਿਧਾਵਾਂ ਸ਼ਾਮਲ ਹਨ।
- ਇਸਨੂੰ ਇਸਦੇ ਪਲੱਗਇਨਜ਼ (plugins) ਦੀ ਸਹਾਇਤਾ ਨਾਲ ਪੂਰੀ ਤਰਾਂ ਕਨਫੀਗਰ (configure) ਅਤੇ ਐਕਸਟੈਂਡ (extend) ਕਰਨ ਯੋਗ ਬਣਾਇਆ ਗਿਆ ਹੈ।

- ਇਹ ਅਜਿਹਾ ਵਰਕਸਪੇਸ ਪ੍ਰਦਾਨ ਕਰਦਾ ਹੈ ਜਿਸ ਵਿੱਚ ਪ੍ਰੋਜੈਕਟਾਂ ਨੂੰ ਇਕੱਠਾ ਕੀਤਾ ਜਾ ਸਕਦਾ ਹੈ।
- ਇਹ Custom memory dump ਅਤੇ syntax highlighting ਦੀ ਸੁਵਿਧਾ ਪ੍ਰਦਾਨ ਕਰਦਾ ਹੈ।
- ਇਹ ਕੋਡ ਅਨੈਲੇਸਿਸ (code analysis) ਆਦਿ ਦੀ ਸਹੂਲਤ ਵੀ ਪ੍ਰਦਾਨ ਕਰਦਾ ਹੈ।

Code::Blocks IDE ਨੂੰ ਹੇਠਾਂ ਦਿੱਤੇ ਲਿੰਕ ਤੋਂ ਡਾਊਨਲੋਡ ਕੀਤਾ ਜਾ ਸਕਦਾ ਹੈ:

http://www.codeblocks.org/downloads

ਬਜ਼ਾਰ ਵਿੱਚ ਉਪਲੱਬਧ ਨਵੇਂ ਜਾਂ ਪੁਰਾਣੇ ਕਿਸੇ ਵੀ C/C++ IDE ਦੀ ਵਰਤੋਂ ਕਰਦੇ ਹੋਏ ਇਸ ਪਾਠ ਵਿੱਚ ਦੱਸੇ ਪ੍ਰੋਗਰਾਮ ਬਣਾਏ ਜਾ ਸਕਦੇ ਹਨ।

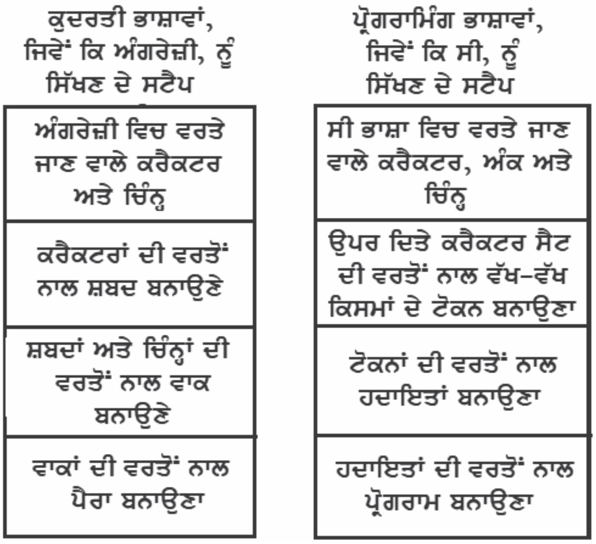
7.4 ਸੀ ਭਾਸ਼ਾ ਦਾ ਪ੍ਰੋਗਰਾਮ ਬਨਾਉਣਾ ਅਤੇ ਚਲਾਉਣਾ (Creating and Executing C Programs)

ਸੀ ਭਾਸ਼ਾ ਦੇ ਪ੍ਰੋਗਰਾਮ ਨੂੰ ਬਨਾਉਣ ਲਈ ਹੇਠ ਲਿਖੇ ਸਟੈਪ ਵਰਤੇ ਜਾ ਸਕਦੇ ਹਨ:

1. ਪ੍ਰੋਗਰਾਮ ਦਾ algorithm ਤਿਆਰ ਕਰੋ।
2. ਕਿਸੇ ਵੀ ਟੈਕਸਟ ਐਡੀਟਰ ਜਾਂ ਸੀ ਭਾਸ਼ਾ ਨੂੰ ਸਪੋਰਟ ਕਰਨ ਵਾਲੇ IDE ਦੀ ਵਰਤੋਂ ਨਾਲ ਸੀ ਭਾਸ਼ਾ ਦਾ ਪ੍ਰੋਗਰਾਮ ਤਿਆਰ ਕਰੋ।
3. ਪ੍ਰੋਗਰਾਮ ਨੂੰ ਫਾਈਲ ਦਾ ਨਾਮ ਦੇਣ ਤੋਂ ਬਾਅਦ C ਐਕਸਟੈਂਸ਼ਨ ਲਿਖਕੇ ਸੇਵ ਕਰੋ।
4. ਪ੍ਰੋਗਰਾਮ ਨੂੰ ਕੰਪਾਈਲ ਕਰੋ।
5. ਜੇਕਰ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਸਿੰਟੈਕਸ ਦੀ ਗਲਤੀ ਹੋਵੇ ਤਾਂ ਉਸਨੂੰ ਠੀਕ ਕਰੋ ਅਤੇ ਸਟੈਪ 4 ਨੂੰ ਦੁਹਰਾਓ।
6. ਪ੍ਰੋਗਰਾਮ ਨੂੰ ਰਨ (Execute) ਕਰੋ।
7. ਆਉਟਪੁੱਟ ਵਿੰਡੋ ਵਿੱਚ ਅਉਟਪੁੱਟ ਨੂੰ ਦੇਖੋ।

7.5 ਸੀ ਨਾਲ ਸ਼ੁਰੂਆਤ ਕਰਨਾ (Getting Sarted with C) :

ਮਨੁੱਖਾਂ ਨਾਲ ਸੰਚਾਰ ਕਰਨ ਲਈ ਅਸੀਂ ਕੁਦਰਤੀ ਭਾਸ਼ਾਵਾਂ, ਜਿਵੇਂ ਕਿ ਅੰਗਰੇਜੀ, ਹਿੰਦੀ ਅਤੇ ਪੰਜਾਬੀ ਆਦਿ ਦੀ ਵਰਤੋਂ ਕਰਦੇ ਹਾਂ। ਪਰ ਕੰਪਿਊਟਰਾਂ ਨਾਲ ਸੰਚਾਰ ਕਰਨ ਲਈ ਅਸੀਂ ਸਿਰਫ ਉਹੀ ਭਾਸ਼ਾਵਾਂ ਦੀ ਵਰਤੋਂ ਕਰ ਸਕਦੇ ਹਾਂ ਜਿਹੜੀਆਂ ਕੰਪਿਊਟਰ ਸਿਸਟਮ ਦੁਆਰਾ ਸਿੱਧੇ ਜਾਂ ਅਸਿੱਧੇ ਢੰਗ ਨਾਲ ਸਮਝੀਆ ਜਾ ਸਕਦੀਆਂ ਹਨ। ਇਹਨਾਂ ਭਾਸ਼ਾਵਾਂ ਨੂੰ ਹੀ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾਵਾਂ ਕਿਹਾ ਜਾਂਦਾ ਹੈ। ਸੀ ਇੱਕ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾ ਹੈ। ਜਿਵੇਂ ਕਿ ਸਾਨੂੰ ਕਿਸੇ ਵੀ ਕੁਦਰਤੀ ਭਾਸ਼ਾ ਦੀ ਵਰਤੋਂ ਕਰਨ ਤੋਂ ਪਹਿਲਾਂ ਇਸਨੂੰ ਸਿੱਖਣਾ ਪੈਂਦਾ ਹੈ, ਠੀਕ ਉਸੇ ਤਰ੍ਹਾਂ ਸਾਨੂੰ ਕੰਪਿਊਟਰਾਂ ਨਾਲ ਸੰਚਾਰ (communication) ਕਰਨ ਲਈ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸਾਵਾਂ ਵੀ ਸਿੱਖਣੀਆਂ ਪੈਂਦੀਆਂ ਹਨ। ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸਾਵਾਂ ਨੂੰ ਸਿੱਖਣਾ ਵੀ ਕਿਸੇ ਵੀ ਕੁਦਰਤੀ ਭਾਸਾ, ਜਿਵੇਂ ਹਿੰਦੀ, ਪੰਜਾਬੀ ਆਦਿ, ਨੂੰ ਸਿੱਖਣ ਵਾਂਗ ਹੀ ਹੁੰਦਾ ਹੈ।



ਚਿੱਤਰ 7.4 ਕੁਦਰਤੀ ਅਤੇ ਕੰਪਿਊਟਰ ਭਾਸ਼ਾਵਾਂ ਨੂੰ ਸਿੱਖਣ ਵਿਚ ਸਮਾਨਤਾ

ਕੁਦਰਤੀ ਭਾਸ਼ਾਵਾਂ (ਜਿਵੇਂ ਕਿ ਅੰਗਰੇਜੀ, ਹਿੰਦੀ, ਪੰਜਾਬੀ, ਆਦਿ) ਅਤੇ ਕੰਪਿਊਟਰ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸਾਵਾਂ (ਜਿਵੇਂ ਕਿ C, C++, JAVA, ਆਦਿ) ਦੇ ਵਿੱਚਕਾਰ ਇੱਕ ਬਹੁਤ ਹੀ ਨਜ਼ਦੀਕੀ ਸਮਾਨਤਾ ਹੈ। ਕਿਸੇ ਵੀ ਕੁਦਰਤੀ ਭਾਸ਼ਾ ਨੂੰ ਸਿੱਖਣ ਦਾ ਰਵਾਇਤੀ ਢੰਗ (ਉਦਾਹਰਣ ਵਜੋਂ ਅੰਗਰੇਜੀ ਭਾਸ਼ਾ) ਇਹ ਹੈ ਕਿ ਸਭ ਤੋਂ ਪਹਿਲਾਂ ਭਾਸ਼ਾ ਵਿੱਚ ਵਰਤੇ ਜਾਣ ਵਾਲੇ ਅੱਖਰਾਂ ਨੂੰ ਸਿੱਖਿਆ ਜਾਂਦਾ ਹੈ, ਫਿਰ ਇਨ੍ਹਾਂ ਅੱਖਰਾਂ ਨੂੰ ਸ਼ਬਦ ਬਣਾਉਣ ਲਈ ਜੋੜਨਾ ਸਿੱਖਿਆ ਜਾਂਦਾ ਹੈ, ਜਿਸ ਦੇ ਨਤੀਜੇ ਵਜੋਂ ਵਾਕਾਂ ਨੂੰ ਜੋੜਿਆ ਜਾਂਦਾ ਹੈ ਅਤੇ ਵਾਕਾਂ ਨੂੰ ਜੋੜ ਕੇ ਪੈਰੇ ਬਣਾਉਣੇ ਸਿੱਖੇ ਜਾਂਦੇ ਹਨ। ਕਿਸੇ ਵੀ ਕੰਪਿਊਟਰ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾ, ਉਦਾਹਰਣ ਲਈ ਸੀ, ਨੂੰ ਵੀ ਇਸੇ ਤਰ੍ਹਾਂ ਹੀ ਆਸਾਨੀ ਨਾਲ ਸਿੱਖਿਆ ਜਾ ਸਕਦਾ ਹੈ।

ਇਸ ਲਈ ਪ੍ਰੋਗਰਾਮਾਂ ਨੂੰ ਕਿਵੇਂ ਲਿਖਣਾ ਹੈ ਇਹ ਸਿੱਧਾ ਸਿੱਖਣ ਦੀ ਬਜਾਏ ਸਾਨੂੰ ਪਹਿਲਾਂ ਇਹ ਪਤਾ ਹੋਣਾ ਚਾਹੀਦਾ ਹੈ ਕਿ ਸੀ ਭਾਸ਼ਾ ਵਿੱਚ ਕਿਹੜੇ ਅੱਖਰ, ਨੰਬਰ ਅਤੇ ਚਿੰਨ੍ਹ ਵਰਤੇ ਜਾਂਦੇ ਹਨ। ਫਿਰ ਇਨ੍ਹਾਂ ਦੀ ਵਰਤੋਂ ਨਾਲ ਵੱਖ-ਵੱਖ ਟੋਕਨਾਂ ਨੂੰ ਕਿਵੇਂ ਬਣਾਇਆ ਜਾਂਦਾ ਹੈ। ਅੰਤ ਵਿੱਚ, ਇਹਨਾਂ ਟੋਕਨਾ ਨੂੰ ਜੋੜ ਕੇ ਹਦਾਇਤਾਂ ਨੂੰ ਕਿਵੇਂ ਬਣਾਇਆ ਜਾਂਦਾ ਹੈ, ਇਹ ਸਿੱਖਿਆ ਜਾਂਦਾ ਹੈ। ਹਦਾਇਤਾਂ ਦਾ ਇੱਕ ਸਮੂਹ, ਆਖਿਰ ਵਿੱਚ, ਇੱਕ ਪ੍ਰੋਗਰਾਮ ਤਿਆਰ ਕਰਨ ਲਈ ਜੋੜਿਆ ਜਾਵੇਗਾ। ਕੁਦਰਤੀ ਅਤੇ ਕੰਪਿਊਟਰ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸਾਵਾਂ ਸਿੱਖਣ ਦੀ ਇਸ ਸਮਾਨਤਾ ਨੂੰ ਚਿੱਤਰ 7.4 ਦੀ ਵਰਤੋਂ ਕਰਕੇ ਦਰਸਾਇਆ ਜਾ ਸਕਦਾ ਹੈ।

7.6 ਕਰੈਕਟਰ ਸੈੱਟ (Character Set) :

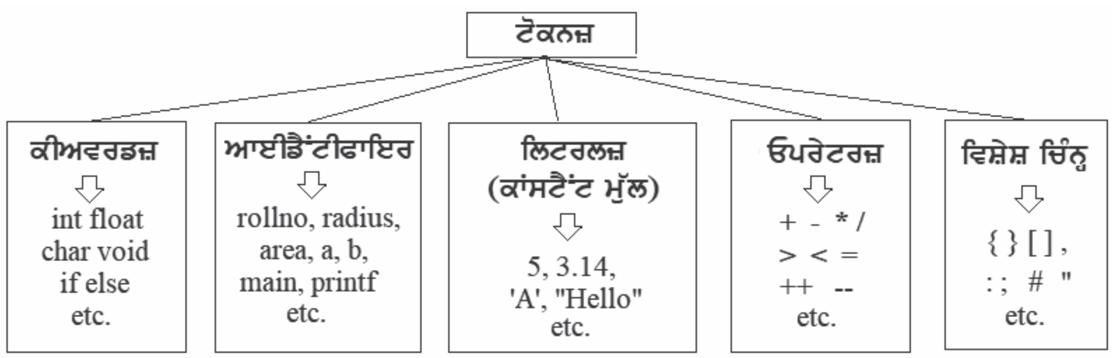
ਇਹ ਕਿਸੇ ਵੀ ਭਾਸ਼ਾ, ਚਾਹੇ ਉਹ ਕੁਦਰਤੀ ਹੋਵੇ ਜਾਂ ਕੰਪਿਊਟਰ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾ, ਨੂੰ ਸਿੱਖਣ ਦਾ ਪਹਿਲਾ ਸਟੈਪ ਹੈ। ਅਸੀਂ ਕੋਈ ਵੀ ਭਾਸਾ ਤਾਂ ਹੀ ਸਿੱਖ ਸਕਦੇ ਹਾਂ ਜੇ ਅਸੀਂ ਉਸ ਭਾਸ਼ਾ ਵਿੱਚ ਵਰਤੇ ਜਾਣ ਵਾਲੇ ਅੱਖਰਾਂ ਅਤੇ ਚਿੰਨ੍ਹਾਂ ਦੀ ਜਾਣਕਾਰੀ ਰੱਖਦੇ ਹਾਂ। ਇਸਲਈ ਸੀ ਭਾਸ਼ਾ ਸਿੱਖਣ ਤੋਂ ਪਹਿਲਾਂ ਸਾਨੂੰ ਇਸ ਵਿੱਚ ਵਰਤੇ ਜਾਣ ਵਾਲੇ ਕਰੈਕਟਰਾਂ ਅਤੇ ਚਿੰਨ੍ਹਾਂ ਤੋਂ ਜਾਣੂ ਹੋਣਾ ਚਾਹੀਦਾ ਹੈ। ਸੀ ਭਾਸ਼ਾ ASCII (ਅਮੇਰੀਕਨ ਸਟੈਂਡਰਡ ਕੋਡ ਫਾਰ ਇਨਫਰਮੇਸਨ ਇੰਟਰਚੇਂਜ) ਕਰੈਕਟਰ ਸੈਟ ਦਾ ਸਮਰਥਨ (supports) ਕਰਦੀ ਹੈ। ਸੀ ਭਾਸ਼ਾ ਵਿੱਚ ਹੇਠਾਂ ਦਿੱਤੇ ਅੱਖਰ ਅਤੇ ਚਿੰਨ੍ਹ ਵਰਤੇ ਜਾ ਸਕਦੇ ਹਨ:

- ਅੰਗਰੇਜ਼ੀ ਭਾਸ਼ਾ ਦੇ ਛੋਟੇ ਵੱਡੇ ਅੱਖਰ (A to Z, a to z)
- ਅੰਕ (0 to 9)
- ਖਾਸ ਚਿੰਨ੍ਹ (ਕੀਅ-ਬੋਰਡ ਉੱਪਰ ਮੌਜੂਦ ਸਾਰੇ ਪ੍ਰਿੰਟੇਬਲ ਚਿੰਨ੍ਹ, ਉਦਾਹਰਨ ਲਈ : ! @ # $ % ^ & * ( ) - _ + = { } [ ] ; : ' " < , > . ? / | \ etc.
- ਕੁੱਝ ਨਾਨ-ਪ੍ਰਿੰਟੇਬਲ ਕਰੈਕਟਰ (ਉਦਾਹਰਣ ਲਈ : \n (new-line character), \t (horizontal-tab character), \a (alert character), ਆਦਿ)

ਉੱਪਰ ਦਿਤੇ ਗਏ ਸਾਰੇ ਕਰੈਕਟਰਾਂ ਅਤੇ ਚਿੰਨ੍ਹਾਂ ਦੇ ਸਮੂਹ ਨੂੰ ਸੀ ਭਾਸ਼ਾ ਦਾ ਕਰੈਕਟਰ ਸੈੱਟ ਕਿਹਾ ਜਾਂਦਾ ਹੈ।

7.7 ਟੋਕਨਜ਼ (Tokens) :

ਟੋਕਨ ਅੰਗਰੇਜੀ ਭਾਸ਼ਾ ਵਿੱਚ ਵਰਤੇ ਜਾਣ ਵਾਲੇ ਸ਼ਬਦਾਂ ਅਤੇ ਵਿਰਾਮ ਚਿੰਨ੍ਹਾਂ ਵਰਗੇ ਹੁੰਦੇ ਹਨ। ਕਿਸੇ ਵੀ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾ ਵਿੱਚ, ਜਿਵੇਂ ਕਿ ਸੀ ਭਾਸ਼ਾ ਵਿੱਚ, ਇੱਕ ਪ੍ਰੋਗਰਾਮ ਟੋਕਨਾਂ ਦਾ ਬਣਿਆ ਹੁੰਦਾ ਹੈ। ਟੋਕਨ ਇੱਕ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਸਭ ਤੋਂ ਛੋਟੀਆਂ ਵਿਅਕਤੀਗਤ ਇਕਾਈਆਂ (smallest individual units) ਹੁੰਦੀਆਂ ਹਨ। ਜਿਵੇਂ ਕਿ ਹੇਠਾਂ ਦਰਸ਼ਾਇਆ ਗਿਆ ਹੈ, ਇੱਕ ਸੀ-ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਪੰਜ ਕਿਸਮਾਂ ਦੇ ਟੋਕਨ ਹੋ ਸਕਦੇ ਹਨ:



ਚਿੱਤਰ 7.5 ਉਦਾਹਰਣਾਂ ਸਹਿਤ ਵੱਖ-ਵੱਖ ਕਿਸਮਾਂ ਦੇ ਟੋਕਨ

ਇਹਨਾਂ ਟੋਕਨਜ਼ ਸੰਬੰਧੀ ਢੁੱਕਵੀਆਂ ਉਦਾਹਰਣਾਂ ਸਹਿਤ ਜਾਣਕਾਰੀ ਹੇਠਾਂ ਦਿੱਤੀ ਗਈ ਹੈ :

7.7.1 ਕੀਵਰਡਜ਼

ਕੀਵਰਡਜ਼ ਨੂੰ ਰਿਜ਼ਰਵ ਵਰਡਜ਼ (reserve words) ਵੀ ਕਿਹਾ ਜਾਂਦਾ ਹੈ। ਇਹ ਸ਼ਬਦ ਸੀ ਭਾਸ਼ਾ ਦੇ ਕੰਪਾਈਲਰ ਵਿੱਚ ਪਹਿਲਾਂ ਤੋਂ ਹੀ ਪਰਿਭਾਸ਼ਿਤ ਹੁੰਦੇ ਹਨ। ਇਨ੍ਹਾਂ ਸਬਦਾਂ ਦਾ ਅਰਥ ਪਹਿਲਾਂ ਤੋਂ ਹੀ ਪਰਿਭਾਸ਼ਿਤ ਕੀਤਾ ਗਿਆ ਹੁੰਦਾ

ਹੈ। ਇਹਨਾਂ ਦੀ ਵਰਤੋਂ ਉਹਨਾਂ ਵਿਸ਼ੇਸ਼ ਉਦੇਸਾਂ ਲਈ ਕੀਤੀ ਜਾਂਦੀ ਹੈ ਜਿਸ ਮੰਤਵ ਲਈ ਉਨ੍ਹਾਂ ਨੂੰ ਪਰਿਭਾਸਤ ਕੀਤਾ ਗਿਆ ਸੀ। ਅਸੀਂ ਇਹਨਾਂ ਦੇ ਅਰਥ ਨਹੀਂ ਬਦਲ ਸਕਦੇ। Turbo C ਵਿੱਚ ਇਹਨਾਂ ਸ਼ਬਦਾਂ ਨੂੰ ਚਿੱਟੇ ਰੰਗ ਵਿੱਚ ਦਿਖਾਇਆ ਜਾਂਦਾ ਹੈ ਜਦੋਂ ਕਿ code::blocks ਵਿੱਚ ਇਹ ਸ਼ਬਦ ਨੀਲੇ ਰੰਗ ਵਿੱਚ ਦਰਸ਼ਾਏ ਜਾਂਦੇ ਹਨ। ਸੀ ਭਾਸਾ ਵਿੱਚ 32 ਕੀਵਰਡ ਹੁੰਦੇ ਹਨ ਪਰ ਨਵੇਂ ਸੀ ਭਾਸ਼ਾ ਕੰਪਾਈਲਰਾਂ ਵਿੱਚ ਕੁਝ ਹੋਰ ਕੀਵਰਡਸ ਵੀ ਸ਼ਾਮਲ ਕੀਤੇ ਗਏ ਹਨ। ਹੇਠਾਂ ਦਿੱਤੀ ਸਾਰਣੀ ਉਹਨਾਂ 32 ਕੀਵਰਡਸ ਦੀ ਸੂਚੀ ਹੈ ਜੋ ਸੀ ਭਾਸ਼ਾ ਦੇ ਸਾਰੇ ਕੰਪਾਈਲਰਾਂ ਵਿੱਚ ਹੁੰਦੇ ਹਨ:

ਟੇਬਲ : 7.1- ਕੀਵਰਡਾਂ ਦੀ ਸੂਚੀ

auto	const	double	float	int	short	struct	unsigned
break	continue	else	for	long	signed	switch	void
case	default	enum	goto	register	sizeof	typedef	volatile
char	do	extern	if	return	static	union	while

ਇਹ ਕੀਵਰਡ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਜਿੱਥੇ ਵੀ ਲੋੜੀਂਦੇ ਹੋਣ ਉਥੇ ਇਹਨਾਂ ਦੀ ਵਰਤੋਂ ਕੀਤੀ ਜਾ ਸਕਦੀ ਹੈ। ਸੀ ਪ੍ਰੋਗਰਾਮਾਂ ਵਿੱਚ ਸਾਰੇ ਕੀਵਰਡ ਅੰਗਰੇਜ਼ੀ ਦੇ ਸਿਰਫ ਛੋਟੇ ਅੱਖਰਾਂ ਵਿੱਚ ਲਿਖੇ ਜਾਂਦੇ ਹਨ। ਕਿਉਂਕਿ ਸੀ-ਭਾਸ਼ਾ ਇੱਕ ਕੇਸ-ਸੰਵੇਦਨਸ਼ੀਲ (case-sensitive) ਭਾਸ਼ਾ ਹੈ, ਇਸ ਲਈ ਜੇ ਅਸੀਂ ਸੀ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਇਹ ਕੀਵਰਡ ਵੱਡੇ ਅੱਖਰ ਵਿੱਚ ਲਿਖਦੇ ਹਾਂ, ਤਾਂ ਸੀ ਭਾਸ਼ਾ ਦਾ ਕੰਪਾਈਲਰ ਗਲਤੀਆਂ ਪ੍ਰਦਰਸ਼ਿਤ ਕਰੇਗਾ। (ਇੱਕ ਕੇਸ-ਸੰਵੇਦਨਸ਼ੀਲ (case sensitive) ਭਾਸ਼ਾ ਇੱਕ ਅਜਿਹੀ ਭਾਸ਼ਾ ਹੈ ਜੋ ਅੰਗਰੇਜ਼ੀ ਦੇ ਛੋਟੇ ਅੱਖਰਾਂ ਅਤੇ ਵੱਡੇ ਅੱਖਰਾਂ ਨੂੰ ਵੱਖ ਵੱਖ ਐਲੀਮੈਂਟਸ ਵਜੋਂ ਸਮਝਦੀ ਹੈ।)

7.7.2 ਆਈਡੈਂਟੀਫਾਇਰਜ਼ (Identifiers) :

ਆਈਡੈਂਟੀਫਾਇਰਜ਼ ਪ੍ਰੋਗਰਾਮ ਦੇ ਐਲੀਮੈਂਟਸ, ਜਿਵੇਂ ਕਿ ਵੇਰੀਏਬਲ, ਕਾਂਸਟੈਂਟਸ, ਐਰੇ, ਫੰਕਸ਼ਨ, ਸਟਰਕਚਰ ਆਦਿ, ਨੂੰ ਦਿੱਤੇ ਨਾਮ ਹਨ। ਹਰ ਪ੍ਰੋਗਰਾਮ ਦੇ ਐਲੀਮੈਂਟ ਨੂੰ ਦੂਜੇ ਐਲੀਮੈਂਟਸ ਨਾਲੋਂ ਵੱਖ ਕਰਨ ਲਈ ਇੱਕ ਨਾਮ ਦਿੱਤਾ ਜਾਣਾ ਜਰੂਰੀ ਹੈ। ਕਿਸੇ ਵੀ ਤੱਤ ਨੂੰ ਦਿੱਤਾ ਗਿਆ ਨਾਮ ਅਰਥਪੂਰਨ ਹੋਣਾ ਚਾਹੀਦਾ ਹੈ ਕਿਉਂਕਿ ਇਸ ਨਾਲ ਪ੍ਰੋਗਰਾਮ ਦੇ ਐਲੀਮੈਂਟਸ ਨੂੰ ਸਮਝਣਾ ਆਸਾਨ ਹੋ ਜਾਂਦਾ ਹੈ। ਪ੍ਰੋਗਰਾਮ ਦੇ ਐਲੀਮੈਂਟਸ ਨੂੰ ਨਾਮ ਦੇਣ ਤੋਂ ਬਾਅਦ ਉਨ੍ਹਾਂ ਨੂੰ ਉਨ੍ਹਾਂ ਦੇ ਨਾਮ ਨਾਲ ਪਛਾਣਿਆ ਜਾ ਸਕਦਾ ਹੈ। ਪ੍ਰੋਗਰਾਮ ਦੇ ਐਲੀਮੈਂਟਸ ਦੇ ਨਾਵਾਂ ਨੂੰ ਪਰਿਭਾਸ਼ਿਤ ਕਰਨ ਲਈ ਸੀ ਪ੍ਰੋਗਰਾਮਾਂ ਵਿੱਚ ਕੁੱਝ ਨਿਯਮਾਂ ਦੀ ਪਾਲਣਾ ਕੀਤੀ ਜਾਂਦੀ ਹੈ। ਐਲੀਮੈਂਟਸ ਦੇ ਨਾਮਕਰਨ ਦੇ ਨਿਯਮ ਹੇਠ ਦਿੱਤੇ ਗਏ ਹਨ:

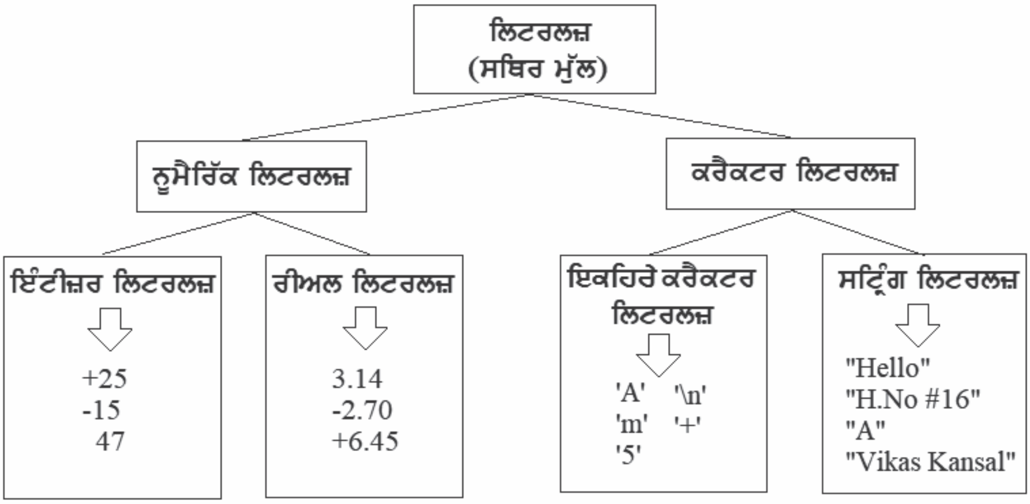
- ਆਈਡੈਂਟੀਫਾਇਰ ਅੱਖਰ ਜਾਂ ਅੰਡਰਸਕੋਰ (__) ਚਿੰਨ੍ਹ ਨਾਲ ਸੁਰੂ ਹੋਣਾ ਚਾਹੀਦਾ ਹੈ। ਇਹ ਕਿਸੇ ਅੰਕ ਦੇ ਨਾਲ ਸੁਰੂ ਨਹੀ ਹੋਣਾ ਚਾਹੀਦਾ। ਉਦਾਹਰਣ ਲਈ: ਆਈਡੈਂਟੀਫਾਇਰ ਨਾਮ 5 Star ਗਲਤ ਹੋਵੇਗਾ ਕਿਉਂਕਿ ਇਹ ਅੰਕ 5 ਨਾਲ ਸੁਰੂ ਹੁੰਦਾ ਹੈ ਜਿਸਦੀ ਆਈਡੈਂਟੀਫਾਇਰ ਦੇ ਨਾਮਕਰਣ ਵਿੱਚ ਆਗਿਆ ਨਹੀਂ ਹੈ।
- ਅੰਡਰਸਕੋਰ (__) ਨੂੰ ਛੱਡ ਕੇ ਕਿਸੇ ਵੀ ਹੋਰ ਵਿਸ਼ੇਸ਼ ਚਿੰਨ੍ਹ ਨੂੰ ਆਈਡੈਂਟੀਫਾਇਰ ਨਾਮ ਵਿੱਚ ਵਰਤਣ ਦੀ ਆਗਿਆ ਨਹੀਂ ਹੈ। ਉਦਾਹਰਣ ਲਈ: ਜੇ ਅਸੀਂ ਇੱਕ ਨਾਮ ਪਰਿਭਾਸਤ ਕਰਦੇ ਹਾਂ: roll#, ਤਾਂ ਇਹ ਗਲਤ ਮੰਨਿਆ ਜਾਵੇਗਾ ਅਤੇ ਕੰਪਾਈਲਰ ਗਲਤੀ ਦਿਖਾਏਗਾ ਕਿਉਂਕਿ ਅਸੀਂ #, ਜੋ ਇੱਕ ਵਿਸੇਸ ਚਿੰਨ੍ਹ ਹੈ, ਨੂੰ ਆਈਡੈਂਟੀਫਾਇਰ ਵਿੱਚ ਵਰਤ ਰਹੇ ਹਾਂ।
- ਆਈਡੈਂਟੀਫਾਇਰ ਨਾਮ ਵਿੱਚ ਲਗਾਤਾਰ ਦੋ ਅੰਡਰਸਕੋਰ ਨਹੀਂ ਵਰਤੇ ਜਾ ਸਕਦੇ। ਉਦਾਹਰਣ ਲਈ: ਆਈਡੈਂਟੀਫਾਇਰ roll_ _ no ਗਲਤ ਹੋਵੇਗਾ ਕਿਉਂਕਿ ਅਸੀਂ ਨਾਮ ਵਿੱਚ ਲਗਾਤਾਰ ਦੋ ਅੰਡਰਸਕੋਰਾਂ ਦੀ ਵਰਤੋਂ ਕੀਤੀ ਹੈ ਜਿਸਦੀ ਸੀ ਪ੍ਰੋਗਰਾਮਾਂ ਵਿੱਚ ਆਗਿਆ ਨਹੀਂ ਹੈ।
- ਕੁਝ ਸੀ ਭਾਸ਼ਾ ਕੰਪਾਈਲਰਾਂ (Turbo C) ਵਿੱਚ, ਆਈਡੈਂਟੀਫਾਇਰ ਦੀ ਲੰਬਾਈ 31 ਅੱਖਰਾਂ ਤੱਕ ਸੀਮਿਤ ਹੈ। ਇਸਦਾ ਅਰਥ ਹੈ ਕਿ ਆਈਡੈਂਟੀਫਾਇਰ ਨਾਮ ਵਿੱਚ ਵੱਧ ਤੋਂ ਵੱਧ 31 ਅੱਖਰ ਅਤੇ ਘੱਟੋ ਘੱਟ 1 ਅੱਖਰ

ਹੋ ਸਕਦੇ ਹਨ। ਜੇ ਅਸੀਂ ਆਈਡੈਂਟੀਫਾਇਰ ਨਾਮ ਵਿੱਚ 31 ਤੋਂ ਵੱਧ ਅੱਖਰਾਂ ਦੀ ਵਰਤੋਂ ਕਰੀਏ ਤਾਂ ਕੰਪਾਈਲਰ ਕੋਈ ਗਲਤੀ ਨਹੀਂ ਦਿਖਾਏਗਾ, ਬਲਕਿ ਕੰਪਾਈਲਰ ਸਿਰਫ ਪਹਿਲੇ 31 ਅੱਖਰਾਂ ਨੂੰ ਆਈਡੈਂਟੀਫਾਇਰ ਨਾਮ ਵਜੋਂ ਵਰਤੇਗਾ ਅਤੇ ਬਾਕੀ ਦੇ ਅੱਖਰਾਂ ਨੂੰ ਨਜ਼ਰਅੰਦਾਜ ਕਰ ਦੇਵੇਗਾ।

- ਕੀਵਰਡਸ ਨੂੰ ਆਈਡੈਂਟੀਫਾਇਰ ਨਾਮ ਵਜੋਂ ਨਹੀਂ ਵਰਤਿਆ ਜਾ ਸਕਦਾ। ਉਦਾਹਰਣ ਲਈ: ਆਈਡੈਂਟੀਫਾਇਰ ਨਾਮ ਪਰਿਭਾਸ਼ਿਤ ਕਰਨ ਲਈ int ਸ਼ਬਦ ਦੀ ਵਰਤੋਂ ਨਹੀਂ ਕੀਤੀ ਜਾ ਸਕਦੀ ਕਿਉਂਕਿ ਇਹ ਇੱਕ ਕੀਵਰਡ ਹੈ ਅਤੇ ਇਸਦਾ ਇੱਕ ਵਿਸ਼ੇਸ਼ ਅਰਥ ਹੈ।
- ਆਈਡੈਂਟੀਫਾਇਰ ਨਾਮ ਕੇਸ-ਸੰਵੇਦਨਸ਼ੀਲ (case-sensitive) ਹੁੰਦੇ ਹਨ। ਇਸਦਾ ਅਰਥ ਹੈ ਲੋਅਰ-ਕੇਸ ਅਤੇ ਅਪਰ-ਕੇਸ ਵਿੱਚ ਲਿਖੇ ਆਈਡੈਂਟੀਫਾਇਰ ਦੇ ਨਾਮ ਵੱਖਰੇ-ਵੱਖਰੇ ਮੰਨੇ ਜਾਂਦੇ ਹਨ। ਇਸ ਲਈ ਸਾਨੂੰ ਆਈਡੈਂਟੀਫਾਇਰ ਨਾਮ ਰੱਖਦੇ ਸਮੇਂ ਲੋਅਰ ਅਤੇ ਅੱਪਰ-ਕੇਸਾਂ ਦਾ ਧਿਆਨ ਰੱਖਣਾ ਚਾਹੀਦਾ ਹੈ। ਉਦਾਹਰਣ ਲਈ: ਸੀ ਪ੍ਰੋਗਰਾਮਾਂ ਵਿੱਚ roll ਅਤੇ ROLL ਦੋ ਵੱਖ-ਵੱਖ ਆਈਡੈਂਟੀਫਾਇਰ ਮੰਨੇ ਜਾਣਗੇ।
- ਆਈਡੈਂਟੀਫਾਇਰ ਨਾਮ ਵਿੱਚ ਖਾਲੀ ਥਾਂ ਛੱਡਣ ਦੀ ਆਗਿਆ ਨਹੀਂ ਹੈ। ਉਦਾਹਰਣ ਲਈ: ਆਈਡੈਂਟੀਫਾਇਰ ਨਾਮ roll no ਗਲਤ ਹੋਵੇਗਾ ਕਿਉਂਕਿ ਇਸ ਵਿੱਚ ਸਬਦ roll ਅਤੇ no ਦੇ ਵਿੱਚਕਾਰ ਖਾਲੀ ਜਗ੍ਹਾ ਛੱਡੀ ਗਈ ਹੈ।

7.7.3 ਲਿਟਰਲਜ਼ (Literals) :

ਲਿਟਰਲਜ਼ ਨੂੰ ਕਾਂਸਟੈਂਟ ਮੁੱਲ ਵੀ ਕਿਹਾ ਜਾਂਦਾ ਹੈ। ਇਹ ਉਹ ਸਥਿਰ ਮੁੱਲ ਹੁੰਦੇ ਹਨ ਜੋ ਆਮ ਤੌਰ ਤੇ ਸੀ-ਪ੍ਰੋਗਰਾਮਾਂ ਵਿੱਚ ਵੇਰੀਏਬਲਾਂ ਵਿਚ ਸਟੋਰ ਕੀਤੇ ਜਾਂਦੇ ਹਨ। ਇਹਨਾਂ ਸਥਿਰ ਮੁੱਲਾਂ ਨੂੰ ਦੋ ਸ਼੍ਰੇਣੀਆਂ ਵਿੱਚ ਸ਼੍ਰੇਣੀਬੱਧ ਕੀਤਾ ਜਾ ਸਕਦਾ ਹੈ: ਨੂਮੈਰਿੱਕ ਅਤੇ ਕਰੈਕਟਰ ਸਥਿਰ ਮੁੱਲ। ਇਹਨਾਂ ਸ਼੍ਰੇਣੀਆਂ ਨੂੰ ਹੇਠਾਂ ਦਿੱਤੇ ਚਿੱਤਰ ਵਿੱਚ ਉਦਾਹਰਣਾਂ ਦੇ ਨਾਲ ਦਰਸ਼ਾਇਆ ਗਿਆ ਹੈ:



ਚਿੱਤਰ 7.6 ਵੱਖ-ਵੱਖ ਕਿਸਮਾਂ ਦੇ ਲਿਟਰਲਜ਼ (ਸਥਿਰ ਮੁੱਲ)

7.7.3.1 ਨੂਮੈਰਿੱਕ ਲਿਟਰਲਜ਼ (Numeric Literals) : ਇਹ ਸੰਖਿਆਤਮਕ (ਨੂਮੈਰਿੱਕ) ਮੁੱਲ ਹਨ, ਜਿਹਨਾਂ ਨੂੰ ਸੰਖਿਆਤਮਕ ਗਣਨਾ ਕਰਨ ਲਈ ਵਰਤਿਆ ਜਾ ਸਕਦਾ ਹੈ। ਇਹ ਦੋ ਕਿਸਮਾਂ ਦੇ ਹੁੰਦੇ ਹਨ:

- **ਪੂਰਨ ਅੰਕ ਲਿਟਰਲਜ਼ (Integer Literals) :** ਇਹ ਬਿਨ੍ਹਾਂ ਦਸ਼ਮਲਵ ਵਾਲੇ ਲਿਟਰਲ ਹੁੰਦੇ ਹਨ। ਇਸ ਵਿੱਚ positive (+) ਜਾਂ negative (−) ਚਿੰਨ੍ਹ ਦੇ ਨਾਲ 0 ਤੋਂ 9 ਦੇ ਅੰਕ ਹੁੰਦੇ ਹਨ। ਜੇ ਸੰਖਿਆ ਨਾਲ ਕੋਈ ਵੀ + ਜਾਂ - ਦਾ ਕੋਈ ਵੀ ਚਿੰਨ੍ਹ ਨਹੀਂ ਹੁੰਦਾ, ਤਾਂ ਇਹ Positive ਪੂਰਨ ਅੰਕ ਮੰਨਿਆ ਜਾਂਦਾ ਹੈ। ਉਦਾਹਰਣ ਲਈ: 56, +26, −96 ਆਦਿ ਪੂਰਨ ਅੰਕ ਹਨ।

- **ਰੀਅਲ ਲਿਟਰਲਜ਼ (Real Literals) :** ਇਹ ਦਸਮਲਵ ਅੰਕਾਂ ਵਾਲੇ ਲਿਟਰਲਜ਼ ਹੁੰਦੇ ਹਨ। ਇਸ ਵਿੱਚ positive (+) ਜਾਂ negative (-) ਚਿੰਨ੍ਹ ਦੇ ਨਾਲ 0 ਤੋਂ 9 ਦੇ ਅੰਕ ਹੁੰਦੇ ਹਨ। ਇਸ ਵਿੱਚ ਇੱਕ ਦਸਮਲਵ ਪੁਆਇੰਟ (.) ਵੀ ਹੁੰਦਾ ਹੈ ਜੋ ਰੀਅਲ ਸੰਖਿਆ ਦੇ ਪੂਰਨ ਅੰਕ ਅਤੇ ਦਸ਼ਮਲਵ ਭਾਗ ਨੂੰ ਵੱਖ ਕਰਦਾ ਹੈ। ਜੇ ਪੂਰਨ ਅੰਕ ਨਾਲ ਕੋਈ ਚਿੰਨ੍ਹ ਨਹੀਂ ਹੁੰਦਾ, ਤਾਂ ਇਸ ਨੂੰ positive ਰੀਅਲ ਲਿਟਰਲ ਮੰਨਿਆ ਜਾਂਦਾ ਹੈ। ਉਦਾਹਰਣ ਲਈ: 3.14, +256.5896, −96.14, 36.00 ਆਦਿ ਰੀਅਲ ਲਿਟਰਲਜ਼ ਹਨ।

7.7.3.2 ਕਰੈਕਟਰ ਲਿਟਰਲਜ਼ (Character Literals) : ਇਹ ਕਰੈਕਟਰ ਮੁੱਲ ਹੁੰਦੇ ਹਨ ਜੋ ਕਿ ਆਮ ਤੌਰ ਤੇ ਗਣਨਾਵਾਂ ਵਿੱਚ ਸ਼ਾਮਲ ਨਹੀਂ ਹੁੰਦੇ। ਕਰੈਕਟਰ ਲਿਟਰਲਜ਼ ਦੋ ਕਿਸਮਾਂ ਦੇ ਹੁੰਦੇ ਹਨ:

- **ਇੱਕਹਿਰੇ ਕਰੈਕਟਰ ਲਿਟਰਲਜ਼ (Single Character Literals) :** ਇਹ ਉਹ ਲਿਟਰਲਜ਼ ਹੁੰਦੇ ਹਨ ਜਿੰਨ੍ਹਾਂ ਵਿੱਚ ਇੱਕ ਇੱਕਹਿਰੇ ਅੱਖਰ ਨੂੰ ਸਿੰਗਲ ਕੋਮਿਆ (single quotes) ਵਿੱਚ ਰੱਖਿਆ ਜਾਂਦਾ ਹੈ। ਇਹਨਾਂ ਲਿਟਰਲਜ਼ ਵਿੱਚ ਇੱਕ ਤੋਂ ਵੱਧ ਪ੍ਰਿੰਟੇਬਲ ਕਰੈਕਟਰਾਂ ਨੂੰ ਰੱਖਣ ਦੀ ਆਗਿਆ ਨਹੀਂ ਹੁੰਦੀ। ਨਾਨ-ਪ੍ਰਿੰਟੇਬਲ ਕਰੈਕਟਰ ਵੀ ਇਸ ਸ੍ਰੇਣੀ ਵਿੱਚ ਆਉਂਦੇ ਹਨ ਹਾਲਾਂਕਿ ਇਨ੍ਹਾਂ ਕਰੈਕਟਰਾਂ ਵਿੱਚ ਦੋ ਚਿੰਨ੍ਹ ਵਰਤੇ ਜਾਂਦੇ ਹਨ, ਉਦਾਹਰਣ ਵਜੋਂ: ਨਵਾਂ ਲਾਈਨ ਅੱਖਰ (\n - ਬੈਕਸਲੈਸ਼ ਅਤੇ ਇੱਕ ਅੱਖਰ n), ਪਰ ਫਿਰ ਵੀ ਉਹਨਾਂ ਨੂੰ ਇੱਕ ਕਰੈਕਟਰ ਮੰਨਿਆ ਜਾਂਦਾ ਹੈ। ਇੱਕਹਿਰੇ ਕਰੈਕਟਰ ਲਿਟਰਲਜ਼ ਦੀਆਂ ਕੁੱਝ ਉਦਾਹਰਣ ਇਸ ਪ੍ਰਕਾਰ ਹਨ: 'A', 'g', '7', '+', '$', '\n', '\t' ਆਦਿ। ਮੁੱਲ 'AB', '45' ਆਦਿ ਇੱਕਹਿਰੇ ਕਰੈਕਟਰ ਲਿਟਰਲਜ਼ ਦੀਆਂ ਸਹੀ ਉਦਾਹਰਣਾਂ ਨਹੀਂ ਹਨ ਕਿਉਂਕਿ ਇਹਨਾਂ ਵਿੱਚ ਇੱਕ ਤੋਂ ਵੱਧ ਅੱਖਰਾਂ ਨੂੰ ਸਿੰਗਲ ਕਾਮਿਆਂ ਵਿੱਚ ਰੱਖਿਆ ਗਿਆ ਹੈ ਜਿਸ ਦੀ ਇੱਕਹਿਰੇ ਕਰੈਕਟਰ ਲਿਟਰਲਜ਼ ਵਿੱਚ ਆਗਿਆ ਨਹੀਂ ਹੁੰਦੀ।
- **ਸਟਰਿੰਗ ਲਿਟਰਲਜ਼ (String Literals) :** ਇਹਨਾਂ ਲਿਟਰਲਜ਼ ਵਿੱਚ ਇੱਕ ਜਾਂ ਇੱਕ ਤੋਂ ਵੱਧ ਕਰੈਕਟਰਾਂ ਨੂੰ ਦੋਹਰੇ ਕਾਮਿਆਂ (double quotes) ਵਿੱਚ ਰੱਖਿਆ ਜਾ ਸਕਦਾ ਹੈ। ਇਹ ਲਿਟਰਲਜ਼ ਅੰਗਰੇਜ਼ੀ ਦੇ ਅੱਖਰਾਂ, ਅੰਕਾਂ, ਖਾਸ ਚਿੰਨ੍ਹਾਂ ਅਤੇ ਖਾਲੀ ਥਾਵਾਂ ਆਦਿ ਦੇ ਸਮੂਹ ਨਾਲ ਬਣੇ ਹੋ ਸਕਦੇ ਹਨ। ਇਹਨਾਂ ਲਿਟਰਲਜ਼ ਦੀਆਂ ਕੁੱਝ ਉਦਾਹਰਣਾ ਇਸ ਪ੍ਰਕਾਰ ਹਨ: "V Kansal", "A", "House#196", "1829" ਆਦਿ।

7.7.4 ਆਪਰੇਟਰਜ਼ (Operators) :

ਆਪਰੇਟਰ ਉਹ ਚਿੰਨ੍ਹ ਹੁੰਦੇ ਹਨ ਜੋ ਕਿ ਕੁੱਝ ਗਣਿਤਕ ਜਾਂ ਲਾਜੀਕਲ ਓਪਰੇਸ਼ਨ ਕਰਨ ਲਈ ਵਰਤੇ ਜਾਂਦੇ ਹਨ। ਉਦਾਹਰਣ ਲਈ: +, −, *, %, ++, −− ਆਦਿ। ਆਪਰੇਟਰ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਮੁੱਲਾਂ/ਵੇਰੀਏਬਲਾਂ ਉੱਪਰ ਕੰਮ ਕਰਨ ਲਈ ਵਰਤੇ ਜਾਂਦੇ ਹਨ। ਇਹਨਾਂ ਮੁੱਲਾਂ/ਵੇਰੀਏਬਲਾਂ ਨੂੰ ਓਪਰੈਂਡਜ਼ (operands) ਕਿਹਾ ਜਾਂਦਾ ਹੈ। ਸੀ ਭਾਸ਼ਾ ਵਿੱਚ ਬਹੁਤ ਸਾਰੇ ਆਪਰੇਟਰਜ਼ ਪਹਿਲਾਂ ਤੋਂ ਹੀ ਬਣੇ ਹੋਏ ਹਨ। ਇਹਨਾਂ ਸਾਰੇ ਓਪਰੇਟਰਾਂ ਨੂੰ ਤਿੰਨ ਵਿਆਪਕ ਸ੍ਰੇਣੀਆਂ ਵਿੱਚ ਵੰਡਿਆ ਜਾ ਸਕਦਾ ਹੈ: ਯੂਨਰੀ (unary), ਬਾਈਨਰੀ (binary), ਅਤੇ ਟਰਨਰੀ (Ternary)

7.7.5 ਵਿਸ਼ੇਸ਼ ਚਿੰਨ੍ਹ (Special Symbols) :

ਇਹਨਾਂ ਚਿੰਨ੍ਹਾਂ ਨੂੰ ਵਿਰਾਮ ਚਿੰਨ੍ਹ ਵਜੋਂ ਵਰਤਿਆ ਜਾਂਦਾ ਹੈ। ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਹਰੇਕ ਚਿੰਨ੍ਹ ਦੀ ਆਪਣੀ ਵੱਖਰੀ ਵਿਸ਼ੇਸ਼ਤਾ ਹੈ। ਹਰ ਇੱਕ ਚਿੰਨ੍ਹ ਦੀ ਵਰਤੋਂ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਕੁਝ ਖਾਸ ਚੀਜ਼ ਦਰਸਾਉਣ ਲਈ ਕੀਤੀ ਜਾਂਦੀ ਹੈ। ਉਦਾਹਰਣ ਦੇ ਤੌਰ ਤੇ: ਸੈਮੀਕਾਲਨ(;) ਚਿੰਨ੍ਹ ਦੀ ਵਰਤੋਂ ਸਟੇਟਮੈਂਟ ਨੂੰ ਖਤਮ ਕਰਨ ਲਈ ਕੀਤੀ ਜਾਂਦੀ ਹੈ, ਕਾਮੇ (,) ਨੂੰ ਸੈਪਰੇਟਰ ਵਜੋਂ ਵਰਤਿਆ ਜਾਂਦਾ ਹੈ, ਫੰਕਸ਼ਨਾਂ ਨੂੰ ਦਰਸ਼ਾਉਣ ਲਈ ਗੋਲ ਬਰੈਕਟ (), ਐਰੇ ਲਈ ਚਕੋਰ ਬਰੈਕਟ [ ], ਸਟੇਟਮੈਂਟਸ ਨੂੰ ਗਰੁੱਪ ਕਰਨ ਲਈ ਘੁੰਡੀਦਾਰ ਬਰੈਕਟ { } ਵਰਤੇ ਜਾਂਦੇ ਹਨ।

7.8 ਵੇਰੀਏਬਲਸ ਅਤੇ ਕਾਂਸਟੈਂਟਸ (Variables and Constants) :

ਇਹ ਦੋਵੇਂ ਹੀ ਪ੍ਰੋਗਰਾਮ ਦੇ ਮਹੱਤਵਪੂਰਣ ਤੱਤ ਹਨ ਜੋ ਪ੍ਰੋਗਰਾਮਾਂ ਵਿੱਚ ਮੁੱਲ ਨੂੰ ਸਟੋਰ ਕਰਨ ਲਈ ਵਰਤੇ ਜਾਂਦੇ ਹਨ। ਦੋਵਾਂ ਹੀ ਐਲੀਮੈਂਟਸ ਨੂੰ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਇੱਕ ਨਾਮ ਅਤੇ ਉਨ੍ਹਾਂ ਵਿੱਚ ਸਟੋਰ ਕੀਤੇ ਜਾਣ ਵਾਲੇ ਮੁੱਲ ਦੀ ਕਿਸਮ ਦਿੱਤੀ ਜਾਂਦੀ ਹੈ। ਪਰ ਇਹਨਾਂ ਦੋਵਾਂ ਵਿੱਚ ਥੋੜਾ ਫਰਕ ਹੈ। **ਵੇਰੀਏਬਲ** ਸਾਨੂੰ ਪ੍ਰੋਗਰਾਮ ਨੂੰ ਰਨ ਕਰਦੇ ਸਮੇਂ ਉਨ੍ਹਾਂ ਦੇ ਮੁੱਲ

ਬਦਲਣ ਦੀ ਆਗਿਆ ਦਿੰਦੇ ਹਨ ਜਦੋਂ ਕਿ **ਕਾਂਸਟੈਂਟਸ** ਇਹ ਆਗਿਆ ਨਹੀਂ ਦਿੰਦੇ। ਇਸਦਾ ਅਰਥ ਇਹ ਹੋਇਆ ਕਿ ਕਾਂਸਟੈਂਟ ਸਥਿਰ ਮੁੱਲ ਹੁੰਦੇ ਹਨ ਜਦੋਂ ਕਿ ਵੇਰੀਏਬਲ ਦੇ ਮੁੱਲ ਪਰਿਵਰਤਨਸ਼ੀਲ (ਬਦਲਣਯੋਗ) ਹੁੰਦੇ ਹਨ। ਸੀ ਭਾਸ਼ਾ ਇੱਕ ਸਟ੍ਰੌਂਗਲੀ ਟਾਈਪਡ (Strogly Typed) ਭਾਸ਼ਾ ਹੈ, ਜਿਸਦਾ ਅਰਥ ਹੈ ਕਿ ਸਾਨੂੰ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਵੇਰੀਏਬਲ ਅਤੇ ਕਾਂਸਟੈਂਟ ਵਰਤਣ ਤੋਂ ਪਹਿਲਾਂ ਇਹਨਾਂ ਨੂੰ ਡਿਕਲੇਅਰ ਕਰਨਾ ਜਰੂਰੀ ਹੁੰਦਾ ਹੈ। ਜੇਕਰ ਅਸੀਂ ਵੇਰੀਏਬਲ ਜਾਂ ਕਾਂਸਟੈਂਟ ਨੂੰ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਡਿਕਲੇਅਰ ਕੀਤੇ ਬਿਨ੍ਹਾਂ ਇਹਨਾਂ ਦੀ ਵਰਤੋਂ ਕਰਦੇ ਹਾਂ ਤਾਂ ਕੰਪਾਈਲਰ ਇੱਕ ਸਿੰਟੈਕਸ-ਗਲਤੀ (Syntax Error) ਦਿਖਾਵੇਗਾ - "Variable not declared" ਇਸ ਲਈ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਹਰ ਇੱਕ ਵੇਰੀਏਬਲ ਅਤੇ ਕਾਂਸਟੈਂਟ ਨੂੰ ਡਿਕਲੇਅਰ ਕੀਤਾ ਜਾਣਾ ਜਰੂਰੀ ਹੈ। ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਵੇਰੀਏਬਲ ਅਤੇ ਕਾਂਸਟੈਂਟ ਨੂੰ ਡਿਕਲੇਅਰ ਕਰਨ ਦੇ ਸਿੰਟੈਕਸ ਹੇਠਾਂ ਦਿੱਤੇ ਗਏ ਹਨ:

ਵੇਰੀਏਬਲ ਡਿਕਲੇਰੇਸ਼ਨ ਦਾ ਸਿੰਟੈਕਸ:

data_type variable_name;

ਇਥੇ **data_type** ਕੰਪਾਈਲਰ ਨੂੰ ਇਹ ਦਸਦੀ ਹੈ ਕਿ ਵੇਰੀਏਬਲ ਵਿੱਚ ਕਿਸ ਕਿਸਮ ਦਾ ਡਾਟਾ ਸਟੋਰ ਹੋਵੇਗਾ ਅਤੇ **variable_ name** ਇੱਕ ਆਈਡੈਂਟੀਫਾਇਰ ਹੁੰਦਾ ਹੈ ਜੋ ਕੰਪਾਈਲਰ ਨੂੰ ਵੇਰੀਏਬਲ ਦੇ ਨਾਮ ਬਾਰੇ ਦੱਸਦਾ ਹੈ। ਇਸ ਨਾਮ ਦੀ ਵਰਤੋਂ ਨਾਲ ਹੀ ਵੇਰੀਏਬਲ ਵਿੱਚ ਸਟੋਰ ਕਿਤੇ ਗਏ ਮੁੱਲ ਨੂੰ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਵਰਤਿਆ ਜਾ ਸਕਦਾ ਹੈ। ਵੇਰੀਏਬਲ ਡਿਕਲੇਰੇਸ਼ਨ ਦੀ ਹੇਠ ਲਿਖੀ ਉਦਾਹਰਣ ਦੇਖੋ:

int roll_no;

ਇਥੇ int ਇੰਟੀਜ਼ਰ ਡਾਟਾ ਟਾਈਪ ਨੂੰ ਦਰਸ਼ਾਉਂਦਾ ਹੈ ਜਦੋਂ ਕਿ **roll_no** ਇੱਕ ਆਈਡੈਂਟੀਫਾਇਰ ਨਾਮ ਹੈ ਜੋ ਕਿ ਵੇਰੀਏਬਲ ਦੇ ਨਾਂ ਵਜੋਂ ਵਰਤਿਆ ਗਿਆ ਹੈ। ਇਸਦਾ ਮਤਲਬ ਇਹ ਹੋਇਆ ਕਿ ਵੇਰੀਏਬਲ **roll_no** ਵਿੱਚ ਕੇਵਲ ਇੰਟੀਜ਼ਰ ਮੁੱਲਾਂ (ਪੂਰਣ ਅੰਕਾਂ) ਨੂੰ ਹੀ ਸਟੋਰ ਕੀਤਾ ਜਾ ਸਕਦਾ ਹੈ। ਜੇਕਰ ਵੇਰੀਏਬਲ ਨੂੰ ਕੋਈ ਮੁੱਲ ਨਾਂ ਦਿਤਾ ਜਾਵੇ ਤਾਂ ਇਸ ਵਿੱਚ ਗਾਰਬੇਜ਼-ਮੁੱਲ (garbage value) ਸਟੋਰ ਹੋ ਜਾਵੇਗਾ। ਜੇਕਰ ਅਸੀਂ ਡਿਕਲੇਰੇਸ਼ਨ ਸਮੇਂ ਹੀ ਇਸ ਵਿੱਚ ਕੋਈ ਮੁੱਲ ਸਟੋਰ ਕਰਵਾ ਦਿੰਦੇ ਹਾਂ ਤਾਂ ਇਸ ਨੂੰ ਵੇਰੀਏਬਲ ਇਨੀਸ਼ੀਅਲਾਈਜੇਸ਼ਨ (variable initialization) ਕਿਹਾ ਜਾਵੇਗਾ।

ਉਦਾਹਰਣ ਲਈ:

int roll_no=5;

ਇਸ ਵਿੱਚ ਸਟੋਰ ਕੀਤੇ ਗਏ ਮੁੱਲ ਨੂੰ ਬਾਅਦ ਵਿੱਚ ਜਦੋਂ ਮਰਜ਼ੀ ਬਦਲਿਆ ਜਾ ਸਕਦਾ ਹੈ ਕਿਉਕਿ ਇੱਕ ਵੇਰੀਏਬਲ ਸਾਨੂੰ ਇਸਦਾ ਮੁੱਲ ਪ੍ਰੋਗਰਾਮ ਚਲਦੇ ਹੋਏ ਕਿਸੇ ਵੀ ਸਮੇਂ ਬਦਲਣ ਦੀ ਆਗਿਆ ਦਿੰਦਾ ਹੈ।

ਕਾਂਸਟੈਂਟ ਡਿਕਲੇਰੇਸ਼ਨ ਦਾ ਸਿੰਟੈਕਸ :

ਇੱਕ ਵੇਰੀਏਬਲ ਨੂੰ ਕਾਂਸਟੈਂਟ ਵਿੱਚ ਤਬਦੀਲ ਕੀਤਾ ਜਾ ਸਕਦਾ ਹੈ। ਵੇਰੀਏਬਲ ਡਿਕਲੇਰੇਸ਼ਨ ਸਮੇਂ ਇਸ ਦੇ ਅੱਗੇ const ਕੀਅਵਰਡ ਦੀ ਵਰਤੋਂ ਕਰਕੇ ਅਤੇ ਇਸਨੂੰ ਇੱਕ ਨਿਸ਼ਚਿੱਤ ਮੁੱਲ ਪ੍ਰਦਾਨ ਕਰਦੇ ਹੋਏ ਕਾਂਸਟੈਂਟ ਵਿੱਚ ਬਦਲਿਆ ਜਾ ਸਕਦਾ ਹੈ। ਉਦਾਹਰਣ ਲਈ ਹੇਠਾਂ ਦਿਤਾ ਸਿੰਟੈਕਸ ਦੇਖੋ:

const data_type constant_name = value;

ਇਥੇ const ਇੱਕ ਕੀਅਵਰਡ ਹੈ ਜੋ ਕੰਪਾਈਲਰ ਨੂੰ ਇਹ ਦਸਦਾ ਹੈ ਕਿ ਇਸਦਾ ਮੁੱਲ ਪ੍ਰੋਗਰਾਮ ਚੱਲਣ ਸਮੇਂ ਨਹੀਂ ਬਦਲਿਆ ਜਾ ਸਕਦਾ। ਕਾਂਸਟੈਂਟ ਨੂੰ ਪਰਿਭਾਸ਼ਿਤ ਕਰਨ ਦੀ ਹੇਠ ਲਿਖੀ ਉਦਾਹਰਣ ਦੇਖੋ:

const float pi=3.14;

ਇਸ ਉਦਾਹਰਣ ਵਿੱਚ ਅਸੀਂ ਇੱਕ ਸਥਿਰ ਮੁੱਲ 3.14 ਨੂੰ ਪਰਿਭਾਸ਼ਿਤ ਕੀਤਾ ਹੈ ਜੋ ਇੱਕ ਰੀਅਲ ਟਾਈਪ ਹੈ। ਇਸਨੂੰ ਇੱਕ ਨਾਮ pi ਦਿਤਾ ਗਿਆ ਹੈ। ਕੀਅਵਰਡ float ਇਹ ਦਸਦਾ ਹੈ ਕਿ ਇਸ ਵਿੱਚ ਰੀਅਲ ਮੁੱਲ ਸਟੋਰ ਹੋਵੇਗਾ ਅਤੇ const ਕੀਅਵਰਡ ਇਸਦੇ ਮੁੱਲ ਨੂੰ ਸਥਿਰ ਬਣਾਉਂਦਾ ਹੈ ਤਾਂ ਕਿ ਇਸਦਾ ਮੁੱਲ ਪ੍ਰੋਗਰਾਮ ਚੱਲਣ ਦੌਰਾਨ ਬਦਲਿਆ ਨਾ ਜਾ ਸਕੇ। ਜੇਕਰ ਅਸੀਂ ਪ੍ਰੋਗਰਾਮ ਚੱਲਣ ਸਮੇਂ ਇਸਦਾ ਮੁੱਲ ਬਦਲਣ ਦੀ ਕੋਸ਼ਿਸ਼ ਕਰਾਂਗੇ ਤਾਂ ਕੰਪਾਈਲਰ ਇੱਕ Error Message ਦਰਸ਼ਾਵੇਗਾ।

7.9 ਡਾਟਾ ਟਾਈਪਸ (Data Types) :

ਡਾਟਾ ਟਾਈਪ ਪਰਿਭਾਸ਼ਿਤ ਕਰਦੀ ਹੈ ਕਿ ਪ੍ਰੋਗਰਾਮ ਦੇ ਤੱਤ, ਜਿਵੇਂ ਕਿ ਵੇਰੀਏਬਲ, ਕਾਂਸਟੈਂਟ, ਐਰੇ ਆਦਿ, ਵਿੱਚ ਕਿਸ ਕਿਸਮ ਦਾ ਡਾਟਾ ਸਟੋਰ ਹੁੰਦਾ ਹੈ। ਡਾਟਾ ਟਾਈਪਸ ਵੇਰੀਏਬਲ ਜਾਂ ਹੋਰ ਪ੍ਰੋਗਰਾਮ ਐਲੀਮੈਂਟਾਂ ਲਈ ਮੁੱਲਾਂ (values) ਦੀ ਇੱਕ ਖਾਸ ਸੀਮਾ (range) ਨੂੰ ਪਰਿਭਾਸ਼ਿਤ ਕਰਦੀਆਂ ਹਨ। ਸੀ ਭਾਸ਼ਾ ਇੱਕ ਸਟ੍ਰਾਂਗਲੀ ਟਾਈਪਡ (strongly typed) ਭਾਸ਼ਾ ਹੈ ਜਿਸ ਕਾਰਣ ਵੇਰੀਏਬਲ ਡਿਕਲੇਅਰ ਕਰਨ ਸਮੇਂ ਇਸਦੇ ਡਾਟਾ ਦੀ ਟਾਈਪ ਨੂੰ ਲਿਖਣਾ ਜਰੂਰੀ ਹੁੰਦਾ ਹੈ।

ਸੀ ਭਾਸ਼ਾ ਕਈ ਤਰ੍ਹਾਂ ਦੇ ਵੱਖੋ-ਵੱਖਰੇ ਡਾਟਾ ਦਾ ਸਮਰਥਨ (support) ਕਰਦੀ ਹੈ। ਇਨ੍ਹਾਂ ਵਿਚੋਂ ਹਰ ਇੱਕ ਡਾਟਾ ਨੂੰ ਕੰਪਿਊਟਰ ਦੀ ਮੈਮਰੀ ਵਿੱਚ ਵੱਖਰੇ-ਵੱਖਰੇ ਤਰੀਕੇ ਨਾਲ ਦਰਸ਼ਾਇਆ ਜਾ ਸਕਦਾ ਹੈ। ਡਾਟਾ ਨੂੰ ਮੈਮਰੀ ਵਿੱਚ ਸਟੋਰ ਕਰਨ ਸੰਬੰਧੀ ਕਦਰਾਂ ਕੀਮਤਾਂ ਇੱਕ ਮਸ਼ੀਨ ਤੋਂ ਦੂਜੀ ਮਸ਼ੀਨ ਅਤੇ ਇੱਕ ਕੰਪਾਈਲਰ ਤੋਂ ਦੂਜੇ ਕੰਪਾਈਲਰ ਵਿੱਚ ਵੱਖੋ-ਵੱਖਰੀਆਂ ਹੋ ਸਕਦੀਆਂ ਹਨ। ਉਦਾਹਰਣ ਲਈ: Turbo C ਵਿੱਚ int ਡਾਟਾ ਟਾਈਪ ਮੈਮਰੀ ਵਿੱਚ ਮੁੱਲ ਸਟੋਰ ਕਰਨ ਲਈ 2 ਬਾਈਟ ਲੈਂਦਾ ਹੈ ਜਦੋਂ ਕਿ Code::Blocks ਵਿੱਚ int ਲਈ 4 ਬਾਈਟ ਮੈਮਰੀ ਦੀ ਜਰੂਰਤ ਪੈਂਦੀ ਹੈ। ਹੇਠ ਦਿੱਤੀ ਸਾਰਣੀ **ਸਟੈਂਡਰਡ** ਸੀ ਭਾਸਾ ਵਿੱਚ ਉਪਲੱਬਧ ਵੱਖ-ਵੱਖ ਮੁਢਲੀਆਂ ਡਾਟਾ ਟਾਈਪਸ ਨੂੰ ਦਰਸ਼ਾ ਰਹੀ ਹੈ:

ਟੇਬਲ 7.2: ਸੀ ਭਾਸ਼ਾ ਵਿੱਚ ਉਪਲਬੱਧ ਮੁਢਲੀਆਂ ਡਾਟਾ ਟਾਈਪਸ

ਕੀਅਵਰਡ	ਵੇਰਵਾ	ਲੋੜੀਂਦੀ ਮੈਮਰੀ	ਮੁੱਲਾਂ ਦੀ ਸੀਮਾ	ਫਾਰਮੇਟ ਕੋਡ
char	ਇੱਕਹਿਰੇ ਕਰੈਕਟਰ ਮੁੱਲਾਂ ਨੂੰ ਸਟੋਰ ਕਰਨ ਲਈ ਵਰਤਿਆ ਜਾਂਦਾ ਹੈ	1 ਬਾਈਟ	-128 ਤੋਂ127	%c
int	ਪੂਰਣ ਅੰਕ ਮੁੱਲਾਂ ਨੂੰ ਸਟੋਰ ਕਰਨ ਲਈ ਵਰਤਿਆ ਜਾਂਦਾ ਹੈ।	2 ਬਾਈਟ	-32768 ਤੋਂ +32767	%d
float	ਸਿੰਗਲ ਪ੍ਰਿਸੀਜ਼ਨ ਫਲੋਟ ਮੁੱਲਾਂ ਨੂੰ ਸਟੋਰ ਕਰਨ ਲਈ ਵਰਤੀ ਜਾਂਦੀ ਹੈ।	4 ਬਾਈਟ	3.4×10^{-38} ਤੋਂ $3.4\times10^{+38}$	%f
double	ਡਬਲ ਪ੍ਰਿਸੀਜ਼ਨ ਫਲੋਟ ਮੁੱਲਾਂ ਨੂੰ ਸਟੋਰ ਕਰਨ ਲਈ ਵਰਤੀ ਜਾਂਦੀ ਹੈ।	8 ਬਾਈਟ	1.7×10^{-308} ਤੋਂ $1.7\times10^{+308}$	%lf
void	ਉਹਨਾਂ ਫੰਕਸ਼ਨਾਂ ਨਾਲ ਵਰਤੀ ਜਾਂਦੀ ਹੈ ਜੋ ਕੋਈ ਮੁੱਲ ਵਾਪਿਸ ਨਹੀਂ ਕਰਦੇ।	-	-	-

7.10 ਸੀ ਭਾਸ਼ਾ ਵਿੱਚ ਹੈਡਰ ਫਾਈਲਾਂ (Header Files in C) :

ਸੀ ਭਾਸ਼ਾ ਪ੍ਰੋਗਰਾਮਾਂ ਵਿੱਚ ਕਈ ਕਿਸਮਾਂ ਦੇ ਕੰਮ ਕਰਨ ਲਈ ਪਹਿਲਾਂ ਤੋਂ ਹੀ ਪਰਿਭਾਸਤਿ ਫੰਕਸਨਾਂ ਦੀ ਇੱਕ ਵਿਸ਼ਾਲ ਲਾਇਬ੍ਰੇਰੀ ਪ੍ਰਦਾਨ ਕਰਦਾ ਹੈ। ਇਹ ਸਾਰੇ ਫੰਕਸ਼ਨ ਲਾਇਬ੍ਰੇਰੀ ਫੰਕਸ਼ਨ ਅਖਵਾਉਂਦੇ ਹਨ। ਇਹਨਾਂ ਫੰਕਸ਼ਨਾਂ ਨੂੰ ਪ੍ਰਬੰਧਿਤ ਕਰਨ ਲਈ ਤਰਕ ਅਨੁਸਾਰ ਗਰੁੱਪਾਂ ਵਿੱਚ ਵੰਡ ਕੇ ਵੱਖ-ਵੱਖ ਫਾਈਲਾਂ ਵਿੱਚ ਸਟੋਰ ਕੀਤਾ ਜਾਂਦਾ ਹੈ। ਇਨ੍ਹਾਂ ਫਾਈਲਾਂ ਨੂੰ ਹੀ ਹੈਡਰ ਫਾਈਲਾਂ ਕਿਹਾ ਜਾਂਦਾ ਹੈ। ਸਾਰੀਆਂ ਹੈਡਰ ਫਾਈਲਾਂ ਦੀ ਐਕਸਟੈਂਸਨ **.h** ਹੁੰਦੀ ਹੈ। ਸਾਨੂੰ ਆਪਣੇ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਇਹਨਾਂ ਫੰਕਸ਼ਨਾਂ ਦੀ ਵਰਤੋਂ ਕਰਨ ਲਈ ਇਹਨਾਂ ਹੈਡਰ ਫਾਈਲਾਂ ਨੂੰ ਸ਼ਾਮਿਲ ਕਰਨਾ ਪੈਂਦਾ ਹੈ। ਇਸ ਕੰਮ ਲਈ #include ਪ੍ਰੀ-ਪ੍ਰੋਸੈਸਰ ਨਿਰਦੇਸ਼ਾਂ ਦੀ ਵਰਤੋਂ ਕੀਤੀ ਜਾਂਦੀ ਹੈ। ਸੀ ਭਾਸ਼ਾ ਵਿੱਚ ਸਾਰੇ ਪ੍ਰੀ-ਪ੍ਰੋਸੈਸਰ ਨਿਰਦੇਸ਼ # ਚਿੰਨ੍ਹ ਨਾਲ ਸੁਰੂ ਹੁੰਦੇ ਹਨ। ਉਦਾਹਰਣ ਲਈ ਹੇਠਾਂ ਦਿੱਤੀ ਉਦਾਹਰਣ ਦੇਖੋ:

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

ਇਸ ਉਦਾਹਰਣ ਵਿੱਚ stdio.h ਇੱਕ ਹੈਡਰ ਫਾਈਲ ਹੈ ਅਤੇ #include ਇੱਕ ਪ੍ਰੀ-ਪ੍ਰੋਸੈਸਰ ਨਿਰਦੇਸ਼ ਹੈ। ਦਿਤੀ ਗਈ ਉਦਾਹਰਣ ਦੀ ਵਰਤੋਂ ਨਾਲ ਅਸੀਂ stdio.h ਹੈਡਰ ਫਾਈਲ ਵਿੱਚ ਮੌਜੂਦ ਕਿਸੇ ਵੀ ਫੰਕਸ਼ਨ ਨੂੰ ਅਪਣੇ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਵਰਤ ਸਕਦੇ ਹਾਂ।

ਸੀ ਭਾਸ਼ਾ ਵਿੱਚ ਬਹੁਤ ਸਾਰੀਆਂ ਹੈਡਰ ਫਾਈਲਾਂ ਮੌਜੂਦ ਹਨ। ਅਸੀਂ ਆਪਣੇ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਕਿਹੜੀਆਂ ਹੈੱਡਰ ਫਾਈਲਾਂ ਵਰਤਣੀਆਂ ਹਨ, ਇਹ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਸਾਡੀ ਜਰੂਰਤ ਤੇ ਨਿਰਭਰ ਕਰਦਾ ਹੈ। ਹੇਠਾਂ ਕੁਝ ਆਮ ਵਰਤੀਆਂ ਜਾਣ ਵਾਲੀਆਂ ਹੈਡਰ ਫਾਈਲਾਂ ਦਿਤੀਆਂ ਗਈਆਂ ਹਨ:

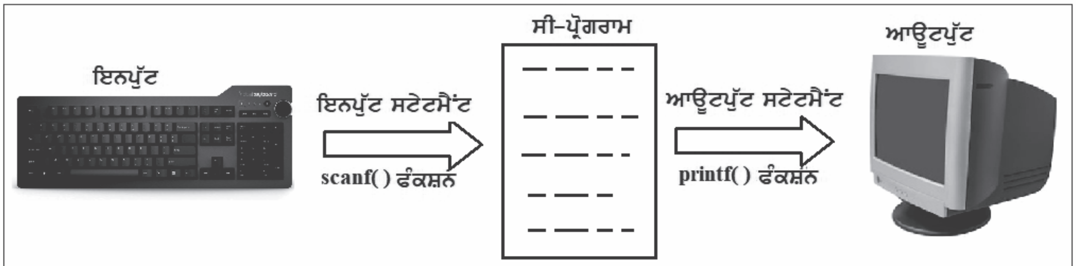
- **ਹੈਡਰ ਫਾਈਲ stdio.h :** ਇਸ ਫਾਈਲ ਦਾ ਪੂਰਾ ਨਾਂ ਸਟੈਂਡਰਡ ਇਨਪੁੱਟ ਆਉਟਪੁੱਟ ਹੈਡਰ ਫਾਈਲ ਹੈ। ਇਸ ਫਾਈਲ ਵਿੱਚ ਉਹ ਫੰਕਸ਼ਨ ਮੌਜੂਦ ਹੁੰਦੇ ਹਨ ਜਿਨ੍ਹਾਂ ਦੀ ਵਰਤੋਂ ਸਟੈਂਡਰਡ ਇਨਪੁੱਟ ਆਉਟਪੁੱਟ ਯੰਤਰਾਂ ਤੋਂ ਇਨਪੁੱਟ ਲੈਣ ਅਤੇ ਆਉਟਪੁੱਟ ਦੇਣ ਲਈ ਕੀਤੀ ਜਾਂਦੀ ਹੈ। ਉਦਾਹਰਣ ਲਈ: scanf( ) ਅਤੇ printf( ) ਫੰਕਸ਼ਨਜ਼।
- **ਹੈਡਰ ਫਾਈਲ conio.h :** ਇਸ ਫਾਈਲ ਦਾ ਪੂਰਾ ਨਾਂ ਕਨਸੋਲ (Console) ਇਨਪੁੱਟ ਆਉਟਪੁੱਟ ਹੈਡਰ ਫਾਈਲ ਹੈ। ਕਨਸੋਲ ਉਹ ਸਕ੍ਰੀਨ ਹੁੰਦੀ ਹੈ ਜਿਥੇ ਸਾਡਾ ਪ੍ਰੋਗਰਾਮ ਰਨ ਹੁੰਦਾ ਹੈ। ਇਸ ਫਾਈਲ ਵਿੱਚ ਉਹ ਫੰਕਸ਼ਨ ਹੁੰਦੇ ਹਨ ਜੋ ਇਨਪੁੱਟ ਆਉਟਪੁੱਟ ਦੌਰਾਨ ਕਨਸੋਲ ਉੱਪਰ ਕੰਮ ਕਰਨ ਲਈ ਵਰਤੇ ਜਾਂਦੇ ਹਨ। ਉਦਾਹਰਣ ਲਈ: **clrscr( ) ਅਤੇ getch( )** ਫੰਕਸ਼ਨਜ਼।
- **ਹੈਡਰ ਫਾਈਲ math.h :** ਇਸ ਫਾਈਲ ਵਿੱਚ ਗਣਿਤਕ ਅਤੇ ਟ੍ਰਿਗਨੋਮੈਟ੍ਰਿਕ ਫੰਕਸ਼ਨ ਹੁੰਦੇ ਹਨ ਜਿਨ੍ਹਾਂ ਦੀ ਵਰਤੋਂ ਪ੍ਰੋਗਰਾਮਾਂ ਵਿੱਚ ਵੱਖ-ਵੱਖ ਗਣਿਤਕ ਕੰਮਾਂ ਨੂੰ ਕਰਨ ਲਈ ਕੀਤੀ ਜਾਂਦੀ ਹੈ। ਉਦਾਹਰਣ ਲਈ: sqrt( ), pow( ), sin( ), cos() ਆਦਿ।
- **ਹੈਡਰ ਫਾਈਲ string.h :** ਇਸ ਫਾਈਲ ਵਿੱਚ ਉਹ ਫੰਕਸ਼ਨ ਮੌਜੂਦ ਹੁੰਦੇ ਹਨ ਜਿਨ੍ਹਾਂ ਦੀ ਵਰਤੋਂ ਸਟ੍ਰਿੰਗ ਡਾਟਾ ਉੱਪਰ ਕੰਮ ਕਰਨ ਲਈ ਕੀਤੀ ਜਾਂਦੀ ਹੈ। ਉਦਾਹਰਣ ਲਈ: strlen( ), strcpy( ), strupr( ), strlwr( ), strcmp( ) ਆਦਿ।

ਕਿਸੇ ਵੀ ਲਾਇਬ੍ਰੇਰੀ ਫੰਕਸਨ ਦੀ ਵਰਤੋਂ ਕਰਨ ਲਈ ਸਾਨੂੰ ਸਾਡੇ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਪ੍ਰੀ-ਪ੍ਰੋਸੈਸਰ ਨਿਰਦੇਸ # ਦੀ ਵਰਤੋਂ ਕਰਦੇ ਹੋਏ ਢੁੱਕਵੀਂ ਹੈਡਰ ਫਾਈਲ ਸ਼ਾਮਿਲ ਕਰਨੀ ਪੈਂਦੀ ਹੈ।

7.11 ਸੀ ਵਿੱਚ ਇਨਪੁੱਟ ਅਤੇ ਆਉਟਪੁੱਟ ਸਟੇਟਮੈਂਟਾਂ (Input and Output Statements in C) :

ਇਨਪੁੱਟ ਅਤੇ ਆਉਟਪੁੱਟ ਸਟੇਟਮੈਂਟਾਂ, ਪ੍ਰੋਗਰਾਮ ਅਤੇ ਯੂਜ਼ਰ ਵਿੱਚਕਾਰ ਆਪਸੀ ਤਾਲਮੇਲ ਪ੍ਰਦਾਨ ਕਰਦੀਆਂ ਹਨ। ਇਨਪੁੱਟ ਸਟੇਟਮੈਂਟ ਦੀ ਵਰਤੋਂ ਕਰਦਿਆਂ ਯੂਜ਼ਰ ਪ੍ਰੋਗਰਾਮ ਨੂੰ ਇਨਪੁੱਟ ਪ੍ਰਦਾਨ ਕਰਦਾ ਹੈ ਅਤੇ ਆਉਟਪੁੱਟ ਸਟੇਟਮੈਂਟ ਦੀ ਵਰਤੋਂ ਕਰਕੇ ਪ੍ਰੋਗਰਾਮ ਯੂਜ਼ਰ ਨੂੰ ਆਉਟਪੁੱਟ ਦਿੰਦਾ ਹੈ।

ਸੀ ਪ੍ਰੋਗਰਾਮਾਂ ਵਿੱਚ ਸਾਰੇ ਇੱਨਪੁਟ ਅਤੇ ਆਉਟਪੁੱਟ ਓਪਰੇਸਨ ਪਹਿਲਾਂ ਤੋਂ ਪਰਿਭਾਸ਼ਿਤ ਫੰਕਸਨਾਂ ਦੀ ਵਰਤੋਂ ਕਰਦੇ ਹੋਏ ਕੀਤੇ ਜਾਂਦੇ ਹਨ। ਹਾਲਾਂਕਿ, ਸੀ ਭਾਸ਼ਾ ਦੀ ਲਾਇਬ੍ਰੇਰੀ ਬਹੁਤ ਸਾਰੇ ਫਾਰਮੈਟਡ ਅਤੇ ਅਨ-ਫਾਰਮੈਟਡ ਇਨਪੁੱਟ/ਆਉਟਪੁੱਟ ਫੰਕਸ਼ਨ ਪ੍ਰਦਾਨ ਕਰਦੀ ਹੈ, ਪਰ ਆਮ ਤੌਰ ਤੇ ਸੀ ਪ੍ਰੋਗਰਾਮਾਂ ਵਿੱਚ ਇਨਪੁੱਟ/ਆਉਟਪੁੱਟ ਓਪਰੇਸ਼ਨਾਂ ਲਈ scanf( ) ਅਤੇ printf( ) ਫਾਰਮੇਟਡ ਫੰਕਸ਼ਨਾਂ ਦੀ ਵਰਤੋਂ ਕੀਤਾ ਜਾਂਦੀ ਹੈ। ਹੇਠਾਂ ਦਿੱਤਾ ਚਿੱਤਰ ਪ੍ਰੋਗਰਾਮਾਂ ਵਿੱਚ ਇਨਪੁੱਟ ਅਤੇ ਆਉਟਪੁੱਟ ਸਟੇਟਮੈਂਟਾਂ ਦੇ ਉਦੇਸ਼ ਨੂੰ ਦਰਸ਼ਾਉਂਦਾ ਹੈ:



ਚਿੱਤਰ: 7.7 ਪ੍ਰੋਗਰਾਮ ਵਿਚ ਇਨਪੁੱਟ ਅਤੇ ਆਊਟਪੁੱਟ ਸਟੇਟਮੈਂਟ ਦਾ ਮੰਤਵ

ਚਿੱਤਰ 7.7 ਪ੍ਰੋਗਰਾਮ ਵਿਚ ਇਨਪੁੱਟ ਅਤੇ ਆਊਟਪੁੱਟ ਸਟੇਟਮੈਂਟ ਦਾ ਮੰਤਵ

ਹੁਣ ਅਸੀਂ ਸੀ ਭਾਸ਼ਾ ਦੇ ਪ੍ਰੋਗਰਾਮਾਂ ਵਿੱਚ ਅਕਸਰ ਵਰਤੇ ਜਾਂਦੇ ਇਹਨਾਂ ਇਨਪੁੱਟ/ਆਉਟਪੁੱਟ ਲਾਈਬ੍ਰੇਰੀ ਫੰਕਸ਼ਨਾਂ : scanf( ) ਅਤੇ printf( ) ਸੰਬੰਧੀ ਚਰਚਾ ਕਰਾਂਗੇ:

ਇਨਪੁੱਟ ਫੰਕਸ਼ਨ scanf( ) (Input Function scanf( )) :

ਸੀ ਭਾਸ਼ਾ ਦੇ ਲਾਇਬ੍ਰੇਰੀ ਫੰਕਸਨ scanf( ) ਦੀ ਵਰਤੋਂ ਕਰਕੇ ਇੱਕ ਸਟੈਂਡਰਡ ਇਨਪੁੱਟ ਡਿਵਾਈਸ ਤੋਂ ਇਨਪੁੱਟ ਡਾਟਾ ਨੂੰ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਦਾਖਲ ਕੀਤਾ ਜਾ ਸਕਦਾ ਹੈ। ਇਹ ਫੰਕਸਨ ਨੂਮੈਰੀਕਲ, ਇੱਕਹਿਰੇ ਕਰੈਕਟਰ ਅਤੇ ਸਟ੍ਰਿੰਗ ਕਿਸੇ ਵੀ ਤਰ੍ਹਾਂ ਦੇ ਮੁੱਲ ਨੂੰ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਦਾਖਲ ਕਰਨ ਲਈ ਵਰਤਿਆ ਜਾ ਸਕਦਾ ਹੈ। ਇਸ ਫੰਕਸਨਾ ਦੀ ਵਰਤੋਂ ਲਈ ਆਮ ਸਿੰਟੈਕਸ ਇਸ ਪ੍ਰਕਾਰ ਹੈ:

```
scanf("format string", &arg1, &arg2, …….., &argn);
```

ਇਸ ਵਿੱਚ format string ਇੱਕ ਸਟ੍ਰਿੰਗ ਨੂੰ ਦਰਸ਼ਾਉਂਦਾ ਹੈ ਜਿਸ ਵਿੱਚ ਆਰਗੂਮੈਂਟਾਂ ਦੀਆਂ ਡਾਟਾ ਟਾਈਪਸ ਅਨੁਸਾਰ ਫਾਰਮੇਟ ਕੋਡਸ (format codes) ਲਿਖੇ ਜਾਂਦੇ ਹਨ ਅਤੇ arg1, arg2, …….., argn ਆਰਗੂਮੈਂਟਸ ਹੁੰਦੇ ਹਨ ਜੋ ਇਨਪੁੱਟ ਕੀਤੀਆਂ ਜਾਣ ਵਾਲੀਆਂ ਅਲੱਗ-ਅਲੱਗ ਡਾਟਾ ਆਈਟਮਾਂ ਨੂੰ ਦਰਸਾਉਂਦੇ ਹਨ। ਆਮ ਤੌਰ ਤੇ, ਆਰਗੂਮੈਂਟਸ ਵੇਰੀਏਬਲਜ਼ ਹੁੰਦੇ ਹਨ ਜੋ ਐਡਰੈਸ ਆਪਰੇਟਰ & ਨਾਲ ਲਿਖੇ ਜਾਂਦੇ ਹਨ। ਇਹ ਐਡਰੈਸ ਆਪਰੇਟਰ & ਮੈਮਰੀ ਵਿੱਚ ਵੇਰੀਏਬਲ ਦੇ ਐਡਰੈਸ ਨੂੰ ਦਰਸਾਉਂਦੇ ਹਨ ਜਿੱਥੇ ਅਸਲ ਵਿੱਚ ਡਾਟਾ ਨੂੰ ਸਟੋਰ ਕੀਤਾ ਜਾਵੇਗਾ। ਫਾਰਮੇਟ ਸਟ੍ਰਿੰਗ ਅਤੇ ਆਰਗੂਮੈਂਟਸ ਨੂੰ ਕੋਮਾ (,) ਆਪਰੇਟਰ ਦੁਆਰਾ ਵੱਖ ਕੀਤਾ ਜਾਂਦਾ ਹੈ। ਇਸ ਤੱਥ ਨੂੰ ਚੰਗੀ ਤਰ੍ਹਾਂ ਸਮਝਣ ਲਈ ਹੇਠਾਂ ਦਿੱਤੀ ਉਦਾਹਰਣ 'ਤੇ ਗੌਰ ਕਰੋ:

```
int a;
float b;
scanf("%d%f",&a,&b);
```

ਇਸ ਵਿੱਚ "%d%f" ਇੱਕ ਫਾਰਮੇਟ ਸਟ੍ਰਿੰਗ ਹੈ ਜੋ ਕਿ ਇਹ ਦਰਸਾਉਂਦਾ ਹੈ ਕਿ ਕੀਅਬੋਰਡ ਤੋਂ ਕਿਸ ਕਿਸਮ ਦਾ ਡਾਟਾ ਪ੍ਰਾਪਤ ਕੀਤਾ ਜਾਵੇਗਾ। ਫਾਰਮੇਟ ਕੋਡ %d ਵੇਰੀਏਬਲ a ਲਈ ਇੰਟੀਜ਼ਰ ਮੁੱਲ (ਪੂਰਣ ਅੰਕ) ਇਨਪੁੱਟ ਕਰਵਾਉਣ ਲਈ ਵਰਤਿਆ ਗਿਆ ਹੈ ਅਤੇ %f ਵੇਰੀਏਬਲ b ਲਈ ਫਲੋਟ ਮੁੱਲ (float value) ਇਨਪੁੱਟ ਕਰਵਾਉਣ ਲਈ ਵਰਤਿਆ ਗਿਆ ਹੈ। ਇਸ ਵਿੱਚ a ਅਤੇ b ਦੋ ਆਰਗੂਮੈਂਟ ਹਨ ਜੋ ਯੂਜ਼ਰ ਵੱਲੋਂ ਮੁੱਲ ਪ੍ਰਾਪਤ ਕਰਨ ਵਾਲੇ ਵੇਰੀਏਬਲਾਂ ਨੂੰ ਦਰਸ਼ਾਉਂਦੇ ਹਨ। ਵੇਰੀਏਬਲ a ਅਤੇ b ਤੋਂ ਪਹਿਲਾਂ ਵਰਤਿਆ ਗਿਆ ਚਿੰਨ੍ਹ & ਇਹ ਦਸਦਾ ਹੈ ਕਿ ਇੰਟੀਜ਼ਰ ਅਤੇ ਫਲੋਟ ਮੁੱਲ a ਅਤੇ b ਵੇਰੀਏਬਲਾਂ ਨੂੰ ਦਿਤੇ ਗਏ ਮੈਮਰੀ ਅਡਰੈਸਾਂ ਉੱਪਰ ਸਟੋਰ ਕੀਤਾ ਜਾਵੇ।

ਆਉਟਪੁੱਟ ਫੰਕਸ਼ਨ printf( ) (Output function printf( )) :

ਸੀ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ printf( ) ਫੰਕਸ਼ਨ ਦੀ ਵਰਤੋਂ ਕਿਸੇ ਵੀ ਜਾਣਕਾਰੀ ਜਾਂ ਮੁੱਲ ਨੂੰ ਮੋਨੀਟਰ (ਆਉਟਪੁੱਟ) ਸਕ੍ਰੀਨ ਉੱਪਰ ਦਰਸਾਉਂਣ ਲਈ ਕੀਤੀ ਜਾਂਦੀ ਹੈ। ਇਸ ਫੰਕਸ਼ਨ ਦੀ ਵਰਤੋਂ ਆਮ ਤੌਰ ਤੇ ਸਾਧਾਰਣ ਟੈਕਸਟ ਮੈਸੇਜ (text message) ਨੂੰ ਮਾਨੀਟਰ (ਆਉਟਪੁੱਟ) ਸਕ੍ਰੀਨ ਉੱਪਰ ਦਰਸਾਉਂਣ ਲਈ ਕੀਤੀ ਜਾਂਦੀ ਹੈ। ਇਸ ਫੰਕਸ਼ਨ ਦੀ ਵਰਤੋਂ ਕਰਨ ਲਈ ਆਮ ਸਿੰਟੈਕਸ ਹੇਠਾਂ ਦਿੱਤਾ ਗਿਆ ਹੈ:

```
printf("simple text message");
```

ਉਦਾਹਰਣ ਲਈ:

```
printf("Hello from C Language");
```

ਇਹ ਫੰਕਸਨ ਮੋਨੀਟਰ ਸਕ੍ਰੀਨ ਤੇ ਵੇਰੀਏਬਲ ਵਿੱਚ ਸਟੋਰ ਕੀਤੇ ਕਿਸੇ ਵੀ ਨੂਮੈਰੀਕਲ, ਸਿੰਗਲ ਕਰੈਕਟਰ ਅਤੇ ਸਟਰਿੰਗ ਮੁੱਲਾਂ ਨੂੰ ਦਰਸਾਉਣ ਲਈ ਵੀ ਵਰਤਿਆ ਜਾ ਸਕਦਾ ਹੈ। ਮੈਮੋਰੀ ਵਿੱਚ ਸਟੋਰ ਕੀਤੇ ਮੁੱਲ ਪ੍ਰਦਰਸ਼ਿਤ ਕਰਨ ਲਈ ਇਸ ਫੰਕਸ਼ਨ ਦਾ ਆਮ ਸਿੰਟੈਕਸ ਇਸ ਪ੍ਰਕਾਰ ਹੈ:

```
printf("format string", arg1, arg2, …….., argn);
```

ਇਸ ਵਿੱਚ format string ਇੱਕ ਸਟ੍ਰਿੰਗ ਨੂੰ ਦਰਸਾਉਂਦਾ ਹੈ ਜਿਸ ਵਿੱਚ ਆਰਗੂਮੈਂਟਾਂ ਦੀਆਂ ਡਾਟਾ ਟਾਈਪਸ ਅਨੁਸਾਰ ਫਾਰਮੇਟ ਕੋਡਸ (format codes) ਲਿਖੇ ਜਾਂਦੇ ਹਨ ਅਤੇ arg1, arg2, …….., argn ਆਰਗੂਮੈਂਟਸ ਹੁੰਦੇ ਹਨ ਜੋ ਆਉਟਪੁੱਟ ਕੀਤੀਆਂ ਜਾਣ ਵਾਲੀਆਂ ਅਲੱਗ-ਅਲੱਗ ਡਾਟਾ ਆਈਟਮਾਂ ਨੂੰ ਦਰਸ਼ਾਉਂਦੇ ਹਨ। ਇਸ ਵਿੱਚ printf( ) ਫੰਕਸ਼ਨ ਮੈਮਰੀ ਵਿਚੋਂ ਆਰਗੂਮੈਂਟਸ ਦੇ ਮੁੱਲਾਂ (values) ਨੂੰ ਪੜ੍ਹਦਾ (read) ਹੈ ਅਤੇ ਮੋਨੀਟਰ (ਆਉਟਪੁੱਟ) ਸਕ੍ਰੀਨ ਉੱਪਰ ਦਰਸਾਉਂਦਾ ਹੈ। ਫਾਰਮੈਟ ਸਟ੍ਰਿੰਗ ਅਤੇ ਆਰਗੂਮੈਂਟਸ ਨੂੰ ਕੋਮਾ (,) ਆਪਰੇਟਰ ਦੁਆਰਾ ਵੱਖ ਕੀਤਾ ਜਾਂਦਾ ਹੈ। ਇਸ ਤੱਥ ਨੂੰ ਚੰਗੀ ਤਰ੍ਹਾਂ ਸਮਝਣ ਲਈ ਹੇਠਾਂ ਦਿੱਤੀ ਉਦਾਹਰਣ 'ਤੇ ਗੌਰ ਕਰੋ:

```
int a=56;
float b=3.14;
printf("%d%f",a,b);
```

ਇਸ ਵਿੱਚ "%d%f" ਇੱਕ ਫਾਰਮੇਟ ਸਟ੍ਰਿੰਗ ਹੈ ਜੋ ਕਿ ਇਹ ਦਰਸਾਉਂਦਾ ਹੈ ਕਿ ਕਿਸ ਕਿਸਮ ਦਾ ਡਾਟਾ ਮੋਨੀਟਰ ਸਕ੍ਰੀਨ ਉੱਪਰ ਦਰਸ਼ਾਇਆ ਜਾਵੇਗਾ। ਫਾਰਮੇਟ ਕੋਡ %d% ਦੀ ਵਰਤੋਂ ਵੇਰੀਏਬਲ a ਵਿੱਚ ਸਟੋਰ ਇੰਟੀਜ਼ਰ ਮੁੱਲ (ਪੂਰਣ ਅੰਕ) ਨੂੰ ਦਰਸਾਉਂਣ ਲਈ ਕੀਤੀ ਗਈ ਹੈ ਅਤੇ %f ਦੀ ਵਰਤੋਂ ਵੇਰੀਏਬਲ b ਵਿੱਚ ਸਟੋਰ ਫਲੋਟ ਮੁੱਲ (float value) ਨੂੰ ਦਰਸਾਉਂਣ ਲਈ ਕੀਤੀ ਗਈ ਹੈ। ਇਸ ਵਿੱਚ a ਅਤੇ b ਦੋ ਆਰਗੂਮੈਂਟ ਹਨ ਜੋ ਯੂਜ਼ਰ ਨੂੰ ਮੋਨੀਟਰ ਸਕ੍ਰੀਨ ਉੱਪਰ ਮੁੱਲ ਦਰਸਾਉਂਣ ਵਾਲੇ ਵੇਰੀਏਬਲਾਂ ਨੂੰ ਦਰਸਾਉਂਦੇ ਹਨ।

ਚਿੱਤਰ 7.8 ਸੀ ਪ੍ਰੋਗਰਾਮਾਂ ਵਿੱਚ scanf( ) ਅਤੇ printf( ) ਫੰਕਸਨਾਂ ਨਾਲ ਕੰਮ ਕਰਨ ਦੀ ਧਾਰਣਾ ਨੂੰ ਦਰਸ਼ਾ ਰਿਹਾ ਹੈ:

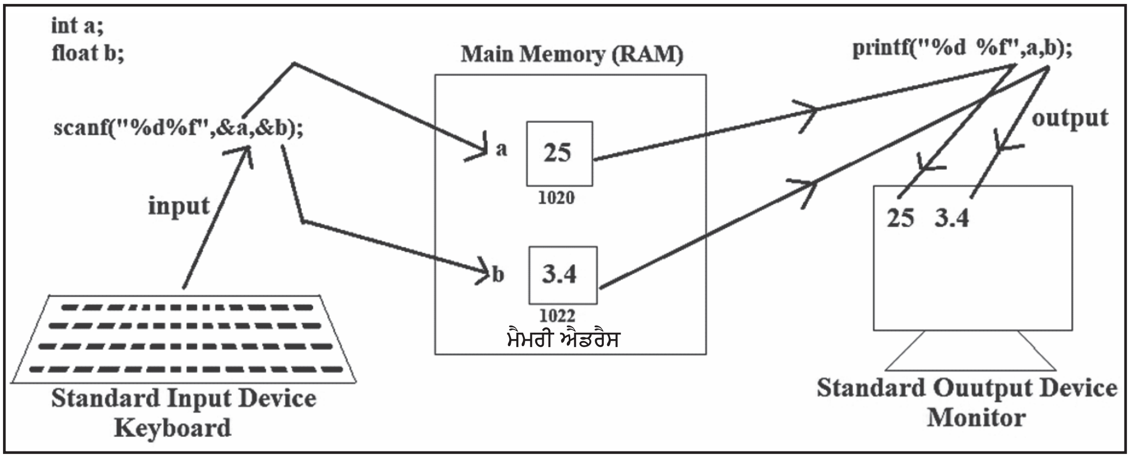


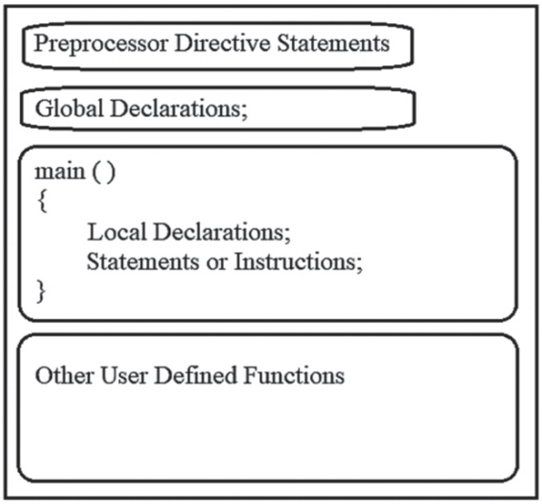
Fig. 7.8 Concept of working with scanf( ) and printf( ) function

ਫੰਕਸਨ scanf( ) ਅਤੇ printf( ) ਸੀ ਭਾਸ਼ਾ ਦੇ ਪ੍ਰੋਗਰਾਮਾਂ ਵਿੱਚ ਵਰਤੇ ਜਾਣ ਵਾਲੇ ਸਭ ਤੋਂ ਵੱਧ ਮਹੱਤਵਪੂਰਨ ਅਤੇ ਲਾਭਦਾਇੱਕ ਫੰਕਸ਼ਨ ਹਨ। ਇਹ ਫੰਕਸਨ stdio.h ਹੈਡਰ ਫਾਈਲ ਦਾ ਹਿੱਸਾ ਹਨ। ਇਸ ਲਈ ਹਰ ਸੀ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਹੈਡਰ ਫਾਈਲ stdio.h ਦੀ ਵਰਤੋਂ ਕੀਤੀ ਜਾਂਦੀ ਹੈ।

7.12 ਸੀ ਭਾਸ਼ਾ ਦੇ ਪ੍ਰੋਗਰਾਮ ਦੀ ਬਣਤਰ (Structure of C Language Program) :

ਜਿਵੇਂ ਕਿ ਅਸੀਂ ਜਾਣਦੇ ਹਾਂ ਕਿ ਇੱਕ ਪ੍ਰੋਗਰਾਮ ਕਿਸੇ ਖਾਸ ਕੰਮ ਨੂੰ ਕਰਨ ਲਈ ਲਿਖੀਆਂ ਗਈਆਂ ਹਦਾਇਤਾਂ ਦਾ ਸਮੂਹ ਹੁੰਦਾ ਹੈ। ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਹਦਾਇਤਾਂ ਦੇ ਲਾਜਿਕਲ ਗਰੁੱਪ ਨੂੰ ਫੰਕਸਨ ਜਾਂ ਬਲਾਕ ਕਿਹਾ ਜਾਂਦਾ ਹੈ। ਹਰੇਕ ਸੀ ਪ੍ਰੋਗਰਾਮ ਇੱਕ ਜਾਂ ਇੱਕ ਤੋਂ ਵੱਧ ਫੰਕਸਨਾਂ ਦਾ ਸੰਗ੍ਰਹਿ ਹੁੰਦਾ ਹੈ ਅਤੇ ਜਿਨ੍ਹਾਂ ਵਿੱਚੋਂ ਕਿਸੇ ਇੱਕ ਫੰਕਸ਼ਨ ਨੂੰ main ਨਾਮ ਦਿਤਾ ਜਾਣਾ ਚਾਹੀਂਦਾ ਹੈ। ਅਜਿਹਾ ਇਸ ਲਈ ਹੋਣਾ ਚਾਹੀਂਦਾ ਹੈ ਕਿਉਂਕਿ main ਫੰਕਸਨ ਸੀ ਪ੍ਰੋਗਰਾਮ ਨੂੰ ਚਲਾਉਣ (execution) ਲਈ ਐਂਟਰੀ ਪੁਆਇੰਟ ਹੁੰਦਾ ਹੈ। ਇਸਦਾ ਅਰਥ ਇਹ ਹੈ ਕਿ ਕੰਪਿਊਟਰ ਸਿਸਟਮ main ਫੰਕਸਨ ਤੋਂ ਸੀ-ਪ੍ਰੋਗਰਾਮ ਨੂੰ ਲਾਗੂ ਕਰਨਾ ਸ਼ੁਰੂ ਕਰਦਾ ਹੈ।

ਹੁਣ ਤੱਕ ਅਸੀਂ ਕਿਸੇ ਵੀ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾ ਨੂੰ ਸਿੱਖਣ ਦੀਆਂ ਬਹੁਤ ਸਾਰੀਆਂ ਧਾਰਨਾਵਾਂ (Concepts) ਤੇ ਚਰਚਾ ਕੀਤੀ ਹੈ। ਕਰੈਕਟਰ-ਸੈੱਟ, ਟੋਕਨਾਂ ਦੀਆਂ ਵੱਖ ਵੱਖ ਕਿਸਮਾਂ, ਵੱਖ-ਵੱਖ ਤਰ੍ਹਾਂ ਦੇ ਡਾਟਾ, ਇਨਪੁੱਟ/ਆਉਟਪੁੱਟ ਸਟੇਟਮੈਂਟਸ ਅਤੇ ਹੈਡਰ ਫਾਈਲਾਂ ਕੀ ਹਨ ਇਹਨਾਂ ਬਾਰੇ ਸਾਨੂੰ ਸਪੱਸ਼ਟ ਹੋ ਚੁੱਕਿਆ ਹੈ। ਇਹ ਸਾਰੀਆਂ ਧਾਰਨਾਵਾਂ ਸੀ ਭਾਸ਼ਾ ਦੀ ਵਰਤੋਂ ਕਰਦੇ ਹੋਏ ਪ੍ਰੋਗਰਾਮਿੰਗ ਸ਼ੁਰੂ ਕਰਨ ਲਈ ਜ਼ਰੂਰੀ ਹਨ। ਹੁਣ ਸਾਨੂੰ ਸੀ-ਪ੍ਰੋਗਰਾਮ ਦੇ ਮੁੱਢਲੇ ਢਾਂਚੇ ਨੂੰ ਜਾਣਨ ਦੀ ਜ਼ਰੂਰਤ ਹੈ ਜਿਸ ਵਿੱਚ ਅਸੀਂ ਇੱਕ ਸਧਾਰਣ ਪ੍ਰੋਗਰਾਮ ਬਣਾਉਣ ਲਈ ਇਹਨਾਂ ਸਾਰੀਆਂ ਚੀਜ਼ਾਂ ਦੀ ਵਰਤੋਂ ਕਰ ਸਕਦੇ ਹਾਂ। ਆਮ ਤੌਰ ਤੇ ਇੱਕ ਸਧਾਰਣ ਚੱਲਣਯੋਗ (executable) ਸੀ-ਪ੍ਰੋਗਰਾਮ ਦੀ ਬਣਤਰ ਚਿੱਤਰ 7.9 ਵਿਚ ਦਿਖਾਏ ਅਨੁਸਾਰ ਹੋ ਸਕਦੀ ਹੈ:



ਚਿੱਤਰ 7.9 ਇੱਕ ਸਾਧਾਰਣ ਚੱਲਣਯੋਗ ਸੀ-ਪ੍ਰੋਗਰਾਮ ਦੀ

ਹਾਲਾਂਕਿ ਅਸੀਂ ਉਪਰੋਕਤ ਸੀ ਭਾਸ਼ਾ ਦੇ ਪ੍ਰੋਗਰਾਮ-ਢਾਂਚੇ ਵਿੱਚ ਵਰਤੀਆਂ ਗਈਆਂ ਕਈ ਧਾਰਨਾਵਾਂ (Concepts) ਬਾਰੇ ਵਿਚਾਰ ਵਟਾਂਦਰਾ ਕਰ ਚੁੱਕੇ ਹਾਂ, ਫਿਰ ਵੀ ਅਸੀਂ ਉਪਰੋਕਤ ਪ੍ਰੋਗਰਾਮ-ਢਾਂਚੇ ਦੇ ਸਾਰੇ ਐਲੀਮੈਂਟਸ ਦੀ ਇੱਕ ਸੰਖੇਪ ਵਿਆਖਿਆ ਕਰਨ ਜਾ ਰਹੇ ਹਾਂ:

- **ਪ੍ਰੀ-ਪ੍ਰੋਸੈਸਰ ਨਿਰਦੇਸ਼ ਸਟੇਟਮੈਂਟਸ (Pre-processor Directive Statements) :** ਪ੍ਰੀ-ਪ੍ਰੋਸੈਸਰ ਸਟੇਟਮੈਂਟਾਂ # ਚਿੰਨ੍ਹ ਨਾਲ ਸ਼ੁਰੂ ਹੁੰਦੀਆਂ ਹਨ। ਇਹ ਸਟੇਟਮੈਂਟਾਂ ਕੰਪਾਈਲਰ ਨੂੰ ਕੰਪਾਈਲੇਸ਼ਨ ਤੋਂ ਪਹਿਲਾਂ ਕੁੱਝ ਓਪਰੇਸ਼ਨ ਕਰਨ ਲਈ ਹਦਾਇਤਾਂ (instructions) ਦਿੰਦੀਆਂ ਹਨ। ਇਨ੍ਹਾਂ ਨਿਰਦੇਸ਼ਕ ਸਟੇਟਮੈਂਟਾਂ ਦੀ ਆਮ ਵਰਤੋਂ ਹੈਡਰ ਫਾਈਲਾਂ ਨੂੰ ਸ਼ਾਮਿਲ (inclusion of header files) ਕਰਨਾ ਜਾਂ ਸਿੰਬੋਲਿਕ ਕਾਂਸਟੈਂਟਾਂ (Symbolic Constants) ਆਦਿ ਨੂੰ ਪਰਿਭਾਸ਼ਿਤ ਕਰਨਾ ਹੈ। ਕੁਝ ਆਮ ਵਰਤੇ ਜਾਣ ਵਾਲੇ ਪ੍ਰੀ-ਪ੍ਰੋਸੈਸਰਾਂ ਦੀਆਂ ਉਦਾਹਰਣਾਂ ਇਸ ਪ੍ਰਕਾਰ ਹਨ:

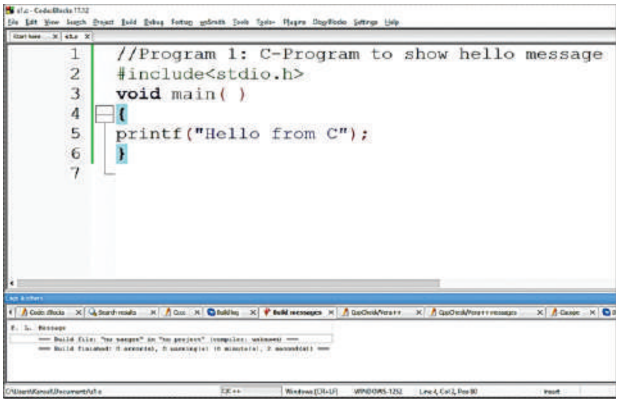
```
#include<stdio.h>
#define PI 3.14
```

- **ਗਲੋਬਲ ਡਿਕਲੇਰੇਸ਼ਨਾਂ (Global Declarations) :** ਇਹਨਾਂ ਡਿਕਲੇਰੇਸ਼ਨਾਂ ਵਿੱਚ ਦਿੱਤੇ ਗਏ ਪ੍ਰੋਗਰਾਮ ਦੇ ਤੱਤ (elements) ਪੂਰੇ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਕਿਤੇ ਵੀ ਵਰਤੇ ਜਾ ਸਕਦੇ ਹਨ। ਇਹ ਡਿਕਲੇਰੇਸ਼ਨਾਂ ਵੇਰੀਏਬਲ, ਫੰਕਸ਼ਨ ਜਾਂ ਕੋਈ ਵੀ ਹੋਰ ਪ੍ਰੋਗਰਾਮ ਦੇ ਤੱਤ (elements) ਹੋ ਸਕਦੇ ਹਨ। ਇਹ ਡਿਕਲੇਰੇਸ਼ਨਾਂ ਫੰਕਸ਼ਨਾਂ ਦੀ ਬਾਡੀ ਤੋਂ ਬਾਹਰ ਲਿਖੀਆਂ ਜਾਂਦੀਆਂ ਹਨ।
- **ਫੰਕਸ਼ਨ main ( ) :** ਸੀ-ਪ੍ਰੋਗਰਾਮ ਦਾ ਲਾਗੂਕਰਣ ਇਸ ਫੰਕਸ਼ਨ ਤੋਂ ਹੀ ਸ਼ੁਰੂ ਹੁੰਦਾ ਹੈ। ਕੋਈ ਵੀ ਸੀ-ਪ੍ਰੋਗਰਾਮ main ਫੰਕਸ਼ਨ ਤੋਂ ਬਿਨ੍ਹਾਂ ਨਹੀਂ ਚਲਾਇਆ ਜਾ ਸਕਦਾ। ਹਰੇਕ ਚਲੱਣਯੋਗ ਸੀ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਇੱਕ main ( ) ਫੰਕਸ਼ਨ ਦਾ ਹੋਣਾ ਲਾਜ਼ਮੀ ਹੁੰਦਾ ਹੈ।
- **ਘੁੰਡੀਦਾਰ ਬਰੈਕਟਾਂ (Braces { }) :** ਘੁੰਡੀਦਾਰ ਬਰੈਕਟਾਂ ਨੂੰ ਸਟੇਟਮੈਂਟਾਂ ਦਾ ਬਲਾਕ ਬਨਾਉਣ ਲਈ ਵਰਤਿਅ ਜਾਂਦਾ ਹੈ। main ( ) ਫੰਕਸ਼ਨ ਤੋਂ ਬਾਅਦ ਖੱਬਾ { ਬਰੈਕਟ ਫੰਕਸ਼ਨ ਦੀ ਸ਼ੁਰੂਆਤ ਅਤੇ ਅੰਤ ਵਿੱਚ ਸੱਜਾ } ਬਰੈਕਟ ਫੰਕਸ਼ਨ ਦੇ ਅੰਤ ਨੂੰ ਦਰਸ਼ਾਉਂਦਾ ਹੈ।
- **ਲੋਕਲ ਡਿਕਲੇਰੇਸ਼ਨਾਂ (Local Declarations) :** ਇੱਕ ਫੰਕਸ਼ਨ ਦੇ ਅੰਦਰ ਕਿਤੀਆਂ ਗਈਆਂ ਡਿਕਲੇਰੇਸ਼ਨਾਂ ਨੂੰ ਲੋਕਲ ਡਿਕਲੇਰੇਸ਼ਨਾਂ ਕਿਹਾ ਜਾਂਦਾ ਹੈ। ਇਹਨਾਂ ਡਿਕਲੇਰੇਸ਼ਨਾਂ ਨੂੰ ਕੇਵਲ ਉਹਨਾਂ ਫੰਕਸ਼ਨਾਂ ਦੇ ਅੰਦਰ ਹੀ ਵਰਤਿਆ ਜਾ ਸਕਦਾ ਹੈ ਜਿਸ ਵਿੱਚ ਇਹਨਾਂ ਨੂੰ ਡਿਕਲੇਅਰ ਕੀਤਾ ਗਿਆ ਹੁੰਦਾ ਹੈ।
- **ਪ੍ਰੋਗਰਾਮ ਸਟੇਟਮੈਂਟਾਂ (Program statements) :** ਇਹ ਸਟੇਟਮੈਂਟਾਂ ਪ੍ਰੋਗਰਾਮ ਦੀਆਂ ਹਦਾਇਤਾਂ ਹੁੰਦੀਆਂ ਹਨ। ਇਹ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਕਿਸੇ ਕੰਮ ਨੂੰ ਕਰਵਾਉਣ ਲਈ ਵਰਤੀਆਂ ਜਾਂਦੀਆਂ ਹਨ। ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਹਰ ਇੱਕ ਚੱਲਣਯੋਗ ਸਟੇਟਮੈਂਟ ਦਾ ਅੰਤ ਸੈਮੀਕਾਲਨ (;) ਨਾਲ ਕੀਤਾ ਜਾਣਾ ਚਾਹੀਦਾ ਹੈ।

- **ਯੂਜ਼ਰ ਡਿਫਾਈਂਡ ਫੰਕਸ਼ਨ (User defined functions) :** ਫੰਕਸ਼ਨ ਸਟੇਟਮੈਂਟਾਂ ਦਾ ਲਾਜਿਕਲ ਗਰੁੱਪ ਹੁੰਦਾ ਹੈ ਜੋ ਇੱਕ ਖਾਸ ਕੰਮ ਕਰਨ ਲਈ ਬਣਾਇਆ ਜਾਂਦਾ ਹੈ। main( ) ਫੰਕਸ਼ਨ ਦੀ ਤਰ੍ਹਾਂ ਅਸੀਂ ਆਪਣੀ ਜਰੂਰਤ ਅਨੁਸਾਰ ਪ੍ਰੋਗਰਾਮ ਵਿਚ ਹੋਰ ਵੀ ਫੰਕਸ਼ਨ ਬਣਾ ਸਕਦੇ ਹੁੰਦੇ ਹਾਂ। ਕਿਉਂਕਿ ਇਹ ਫੰਕਸ਼ਨ ਯੂਜ਼ਰ ਦੁਆਰਾ ਲਿਖੇ ਜਾਂਦੇ ਹਨ ਇਸਲਈ ਇਹਨਾਂ ਫੰਕਸ਼ਨਾਂ ਨੂੰ ਯੂਜ਼ਰ ਡਿਫਾਈਂਡ ਫੰਕਸ਼ਨ ਕਿਹਾ ਜਾਂਦਾ ਹੈ।

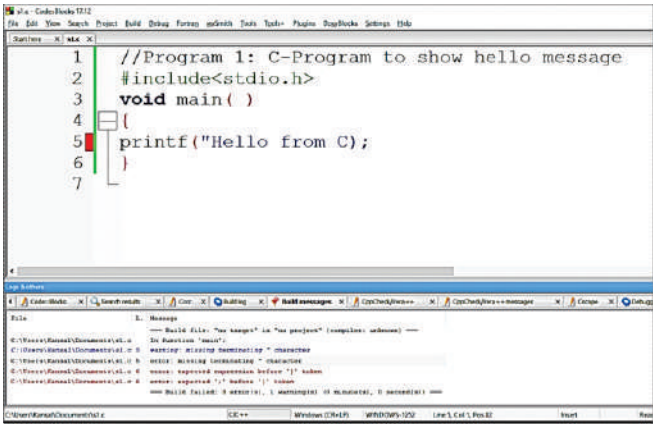
ਆਓ ਹੁਣ ਅਸੀਂ ਇਸ ਪਾਠ ਵਿੱਚ ਜੋ ਵੀ ਸਿੱਖਿਆ ਹੈ ਉਸ ਦੀ ਵਰਤੋਂ ਕਰਦਿਆਂ ਛੋਟੇ-ਛੋਟੇ ਪ੍ਰੋਗਰਾਮ ਬਣਾ ਕੇ ਸੀ ਭਾਸ਼ਾ ਵਿੱਚ ਪ੍ਰੋਗਰਾਮਿੰਗ ਦੀ ਸ਼ੁਰੂਆਤ ਕਰੀਏ। ਅੱਗੇ ਵਧਣ ਤੋਂ ਪਹਿਲਾਂ ਅਸੀਂ ਇਹ ਮੰਨ ਕੇ ਚੱਲ ਰਹੇ ਹਾਂ ਕਿ ਤੁਸੀਂ ਸੀ ਪ੍ਰੋਗਰਾਮਾਂ ਨੂੰ ਬਨਾਉਣ ਲਈ Turbo C ਜਾਂ Code::Blocks ਸਾਫਟਵੇਅਰਾਂ ਨੂੰ ਡਾਊਨਲੋਡ ਕਰਕੇ ਇਨਸਟਾਲ ਕਰ ਚੁੱਕੇ ਹੋ। ਹੁਣ ਅਸੀਂ Code::Blocks ਦੀ ਵਰਤੋਂ ਕਰਦੇ ਹੋਏ ਸੀ-ਭਾਸ਼ਾ ਵਿਚ ਪ੍ਰੋਗਰਾਮ ਬਨਾਉਣ ਸੰਬੰਧੀ ਜਾਣਕਾਰੀ ਹਾਸਲ ਕਰਾਂਗੇ:

1. Code::Blocks ਖੋਲਣ ਲਈ ਡੈਸਕਟਾਪ ਉੱਪਰ ਮੌਜੂਦ ਇਸਦੇ ਆਈਕਾਨ ਉੱਪਰ ਡਬਲ ਕਲਿੱਕ ਕਰੋ।
2. **File → New → Empty File ਜਾਂ ਸ਼ਾਰਟਕੱਟ ਕੀਅ Ctrl+Shift+N** ਦੀ ਵਰਤੋਂ ਕਰਦੇ ਹੋਏ ਨਵੀਂ ਫਾਈਲ ਬਣਾਓ।
3. ਹੁਣ ਹੇਠਾਂ ਦਿੱਤੇ ਚਿੱਤਰ ਵਿਚ ਦਿਖਾਏ ਅਨੁਸਾਰ ਨਵੀਂ ਫਾਈਲ ਵਿਚ ਸੀ ਭਾਸ਼ਾ ਦਾ ਪ੍ਰੋਗਰਾਮ ਟਾਈਪ ਕਰੋ।



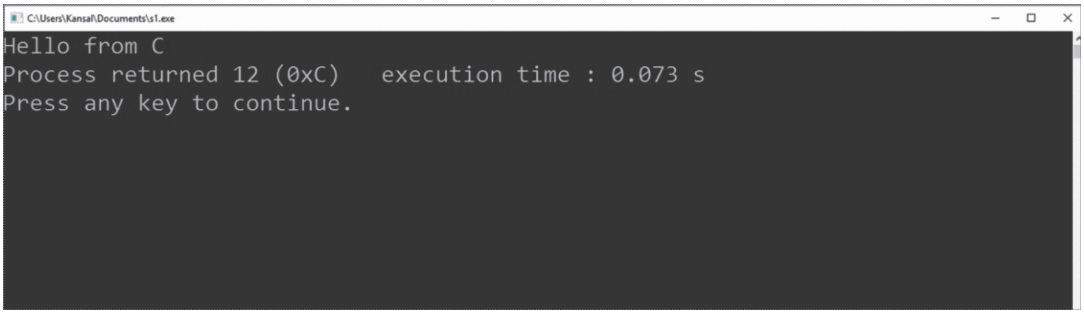
ਪ੍ਰੋਗਰਾਮ : 7.1

4. ਪ੍ਰੋਗਰਾਮ ਦਾ ਸੋਰਸ ਕੋਡ ਟਾਈਪ ਕਰਨ ਤੋਂ ਬਾਅਦ **File → Save File** ਜਾਂ ਸ਼ਾਰਟਕੱਟ ਕੀਅ **Ctrl+S** ਦੀ ਵਰਤੋਂ ਕਰਦੇ ਹੋਏ ਫਾਈਲ ਨੂੰ ਸੇਵ ਕਰੋ। ਫਾਈਲ ਸੇਵ ਕਰਦੇ ਸਮੇਂ ਫਾਈਲ ਦੇ ਨਾਮ ਨਾਲ ਇਸਦੀ ਐਕਸਟੈਂਸ਼ਨ .c ਟਾਈਪ ਕਰਨਾ ਨਾਂ ਭੁੱਲੋ, ਉਦਾਹਰਣ ਲਈ : test1.c, ਇਸ ਵਿਚ test1 ਫਾਈਲ ਦਾ ਨਾਂ ਅਤੇ .c ਸੀ ਭਾਸ਼ਾ ਦੇ ਪ੍ਰੋਗਰਾਮ ਦੀ ਐਕਸਟੈਂਸ਼ਨ ਹੈ।
5. ਹੁਣ **Build → Build and Run** ਜਾਂ ਸ਼ਾਰਟਕੱਟ ਕੀਅ **F9** ਦੀ ਵਰਤੋਂ ਕਰਦੇ ਹੋਏ ਸੋਰਸ ਕੋਡ ਨੂੰ ਕੰਪਾਈਲ ਅਤੇ ਰਨ ਕਰੋ।
6. ਜੇਕਰ ਪ੍ਰੋਗਰਾਮ ਵਿਚ ਗਲਤੀਆਂ ਹੋਣ ਤਾਂ ਇਹ ਗਲਤੀਆਂ ਪ੍ਰੋਗਰਾਮ ਵਿੰਡੋ ਦੇ ਹੇਠਲੇ ਪਾਸੇ ਮੌਜੂਦ Logs and Others ਵਿੰਡੋ ਦੇ Build Messages ਟੈਬ ਵਿਚ ਨਜ਼ਰ ਆਉਣਗੀਆਂ। ਜੇਕਰ Logs and Others ਵਿੰਡੋ ਦਿਖਾਈ ਨਾਂ ਦੇ ਰਹੀ ਹੋਵੇ ਤਾਂ View → Logs ਜਾਂ ਸ਼ਾਰਟਕੱਟ ਕੀਅ F2 ਦੀ ਵਰਤੋਂ ਨਾਲ ਇਸ ਵਿੰਡੋ ਨੂੰ ਖੋਲਿਆ ਜਾ ਸਕਦਾ ਹੈ। ਇਸ ਵਿੰਡੋ ਵਿਚ ਦਿਖਾਈ ਦੇ ਰਹੀਆਂ ਗਲਤੀਆਂ ਨੂੰ ਠੀਕ ਕਰਨ ਉਪਰੰਤ ਹੀ ਸੀ ਭਾਸ਼ਾ ਦੇ ਪ੍ਰੋਗਰਾਮ ਨੂੰ ਕੰਪਾਈਲ ਅਤੇ ਰਨ ਕੀਤਾ ਜਾ ਸਕਦਾ ਹੈ। ਉਦਾਹਰਣ ਲਈ ਹੇਠਾਂ ਦਿਤੇ ਚਿੱਤਰ ਵਿਚ ਲਾਈਨ ਨੰ. 5 ਉੱਪਰ ਗਲਤੀਆਂ ਦਰਸ਼ਾਈਆਂ ਗਈਆਂ ਹਨ:



ਉੱਪਰ ਦਿਤੀ ਤਸਵੀਰ ਵਿਚ ਲਾਈਨ ਨੰ. 5 ਉੱਪਰ ਇਕ ਲਾਲ ਰੰਗ ਦਾ ਬਾਕਸ ਦਿਖਾਈ ਦੇ ਰਿਹਾ ਹੈ ਜੋ ਉਸ ਲਾਈਨ ਵਿਚ ਗਲਤੀ ਨੂੰ ਦਰਸ਼ਾ ਰਿਹਾ ਹੈ। ਗਲਤੀਆਂ ਸੰਬੰਧੀ ਜਾਣਕਾਰੀ ਹੇਠਾਂ Logs and Others ਵਿੰਡੋ ਦੇ Build Messages ਟੈਬ ਵਿਚ ਦੇਖੀ ਜਾ ਸਕਦੀ ਹੈ। ਪ੍ਰੋਗਰਾਮ ਨੂੰ ਕੰਪਾਈਲ ਅਤੇ ਰਨ ਕਰਨ ਤੋਂ ਪਹਿਲਾਂ ਇਹਨਾਂ ਗਲਤੀਆਂ ਨੂੰ ਠੀਕ ਕਰਨਾ ਜ਼ਰੂਰੀ ਹੁੰਦਾ ਹੈ।

7. ਜਦੋਂ ਪ੍ਰੋਗਰਾਮ ਵਿਚ ਗਲਤੀਆਂ ਖਤਮ ਹੋਣ ਤੋਂ ਬਾਅਦ ਇਸਨੂੰ ਕੰਪਾਈਲ ਅਤੇ ਰਨ ਕੀਤਾ ਜਾਵੇਗਾ ਤਾਂ ਪ੍ਰੋਗਰਾਮ ਦੀ ਆਉਟਪੁੱਟ ਸਾਨੂੰ ਹੇਠ ਦਿੱਤੇ ਚਿੱਤਰ ਅਨੁਸਾਰ ਦਿਖਾਈ ਦੇਵੇਗੀ :



```
C:\Users\Kansal\Documents\s1.exe
Hello from C
Process returned 12 (0xC)   execution time : 0.073 s
Press any key to continue.
```

ਪ੍ਰੋਗਰਾਮ 7.1 ਦੀ ਆਉਟਪੁੱਟ

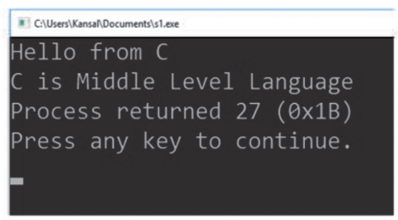
ਇਹ ਅਉਟਪੁੱਟ ਵਿੰਡੋ ਸਾਨੂੰ ਪ੍ਰੋਗਰਾਮ ਨੂੰ ਰਨ ਕਰਨ ਵਿਚ ਲੱਗਣ ਵਾਲਾ ਸਮਾਂ ਵੀ ਦਰਸਾਉਂਦੀ ਹੈ। ਸੋਰਸ ਕੋਡ ਵਿੰਡੋ ਤੇ ਵਾਪਿਸ ਜਾਣ ਲਈ ਕੀਅਬੋਰਡ ਤੋਂ ਕੋਈ ਵੀ ਕੀਅ ਦਬਾਈ ਜਾ ਸਕਦੀ ਹੈ।

ਉੱਪਰ ਦਿਤੇ ਸਟੈਪਾਂ ਦੀ ਵਰਤੋਂ ਕਰਦੇ ਹੋਏ ਅਸੀਂ Code::Blocks ਦੀ ਵਰਤੋਂ ਨਾਲ ਸੀ ਭਾਸ਼ਾ ਵਿਚ ਪ੍ਰੋਗਰਾਮ ਬਣਾ ਕੇ ਉਹਨਾਂ ਨੂੰ ਕੰਪਾਈਲ ਅਤੇ ਰਨ ਕਰ ਸਕਦੇ ਹਾਂ। ਇਸ ਪਾਠ ਵਿਚ ਬਣਾਏ ਗਏ ਪ੍ਰੋਗਰਾਮਾਂ ਦਾ ਮਕਸਦ ਇਸ ਪਾਠ ਵਿਚ ਪੜ੍ਹੇ ਗਏ ਵੱਖ-ਵੱਖ ਸੰਕਲਪਾਂ ਦੀ ਵਰਤੋਂ ਨੂੰ ਦਰਸਾਉਣਾ ਹੈ।

ਪ੍ਰੋਗਰਾਮ 7.2: ਕਈ ਲਾਈਨਾਂ ਵਿੱਚ ਮੈਸੇਜ ਦਰਸਾਉਣ ਲਈ ਸੀ-ਪ੍ਰੋਗਰਾਮ

```c
#include<stdio.h>
void main( )
{
printf("Hello from C");
printf("\nC is Middle Level Language");
}
```

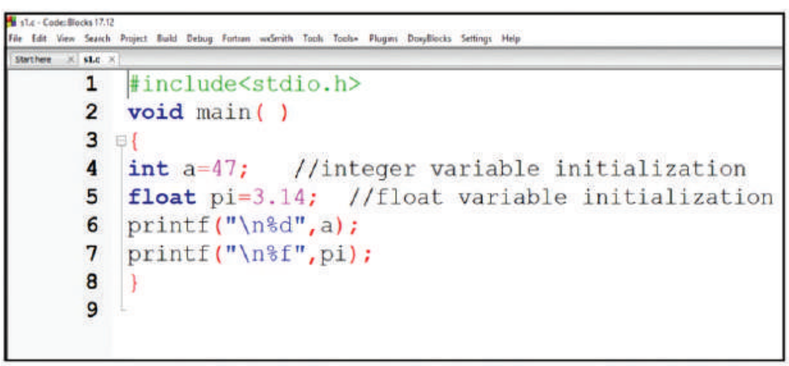
Program 7.2



```
C:\Users\Kansal\Documents\s1.exe
Hello from C
C is Middle Level Language
Process returned 27 (0x1B)
Press any key to continue.
```

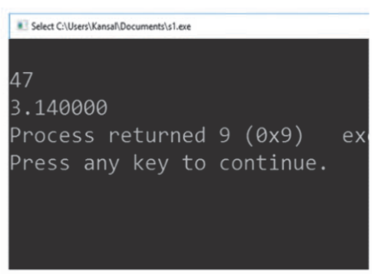
Output of Program 7.2

ਪ੍ਰੋਗਰਾਮ 7.3: ਵੇਰੀਏਬਲ initialization ਅਤੇ ਇਸਦਾ ਮੁੱਲ ਦਰਸਾਉਣ ਲਈ ਸੀ-ਪ੍ਰੋਗਰਾਮ



```c
#include<stdio.h>
void main( )
{
int a=47;    //integer variable initialization
float pi=3.14;   //float variable initialization
printf("\n%d",a);
printf("\n%f",pi);
}
```

Program: 7.3



```
Select C:\Users\Kansal\Documents\s1.exe

47
3.140000
Process returned 9 (0x9)   ex
Press any key to continue.
```

Output of Program: 7.3

ਇਸ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਅਸੀਂ ਕਈ ਕਿਸਮਾਂ ਦੇ ਟੋਕਨ ਵਰਤੇ ਗਏ ਹਨ। ਆਓ ਹੁਣ ਅਸੀਂ ਇਸ ਪ੍ਰੋਗਰਾਮ ਦੀ ਵਰਤੋਂ ਕਰਦੇ ਹੋਏ ਉਹਨਾਂ ਟੋਕਨਾਂ ਬਾਰੇ ਵਿਚਾਰ ਕਰੀਏ ਜੋ ਅਸੀਂ ਇਸ ਪਾਠ ਵਿੱਚ ਪੜ੍ਹੇ ਹਨ:

ਪ੍ਰੋਗਰਾਮ ਦੇ ਸਾਰੇ ਟੋਕਨ : # include <> stdio.h void main ( ) { int a = 47 ; float pi 3.14 printf " " %d %f pi }

ਕੀਅਵਰਡਸ : void, int, float (representing data types)

ਆਈਡੈਂਟੀਫਾਇਰਸ : stdio (ਹੈਡਰ ਫਾਈਲ ਦਾ ਨਾਂ), main, printf (ਫੰਕਸ਼ਨਾਂ ਦੇ ਨਾਂ), a, pi (ਵੇਰੀਏਬਲਾਂ ਦੇ ਨਾਂ)

ਲਿਟਰਲਜ਼ : 47, 3.14 (ਸਥਿੱਰ ਮੁੱਲ)

ਆਪਰੇਟਰਜ਼ : = (ਅਸਾਈਨਮੈਂਟ ਆਪਰੇਟਰ - ਮੁੱਲ ਸਟੋਰ ਕਰਵਾਉਣ ਲਈ)

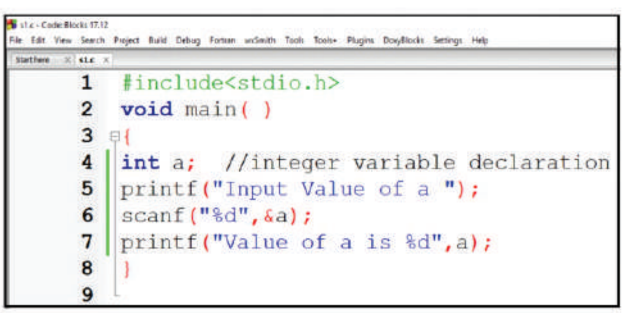
ਵਿਸ਼ੇਸ਼ ਚਿੰਨ੍ਹ : # <> . ( ) { ; " " % }

ਹੁਣ ਅਸੀਂ ਪੂਰੇ ਪ੍ਰੋਗਰਾਮ (ਪ੍ਰੋਗਰਾਮ 3) ਨੂੰ ਲਾਈਨ ਦਰ ਲਾਈਨ ਸਮਝਣ ਜਾ ਰਹੇ ਹਾਂ ਤਾਂ ਜੋ ਸਾਨੂੰ ਪ੍ਰੋਗਰਾਮਿੰਗ ਦੀ ਬਿਹਤਰ ਸਮਝ ਆ ਸਕੇ:

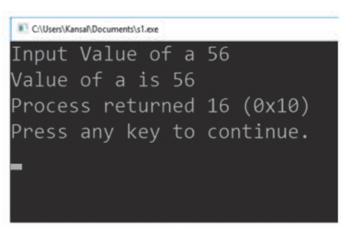
- ਪ੍ਰੋਗਰਾਮ ਦੀ ਲਾਈਨ ਨੰ 1 ਵਿੱਚ, ਅਸੀਂ ਹੈਡਰ ਫਾਈਲ stdio.h ਨੂੰ ਆਪਣੇ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਸ਼ਾਮਿਲ ਕਰਨ ਲਈ ਪ੍ਰੀ-ਪ੍ਰੋਸੈਸਰ ਦੀ ਵਰਤੋਂ ਕੀਤੀ ਹੈ ਤਾਂ ਜੋ ਅਸੀਂ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ printf( ) ਫੰਕਸ਼ਨ ਦੀ ਵਰਤੋਂ ਕਰ ਸਕੀਏ।
- ਲਾਈਨ ਨੰ 2 ਵਿੱਚ, ਅਸੀਂ main( ) ਫੰਕਸ਼ਨ ਦੀ ਸ਼ੁਰੂਆਤ ਕੀਤੀ ਹੈ। ਸਾਡਾ ਪ੍ਰੋਗਰਾਮ ਇਸ main( ) ਫੰਕਸ਼ਨ ਤੋਂ ਚੱਲਣਾ ਸ਼ੁਰੂ ਕਰਦਾ ਹੈ। ਹਰੇਕ ਚਲੱਣਯੋਗ ਸੀ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ main( ) ਫੰਕਸ਼ਨ ਦਾ ਹੋਣਾ ਲਾਜ਼ਮੀ ਹੁੰਦਾ ਹੈ। ਅਸੀਂ ਇੱਕ ਸੀ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਇੱਕ ਤੋਂ ਵੱਧ main( ) ਫੰਕਸ਼ਨ ਨਹੀਂ ਬਣਾ ਸਕਦੇ।
- ਲਾਈਨ ਨੰ 3 ਵਿੱਚ, ਅਸੀਂ ਘੁੰਡੀਦਾਰ ਬਰੈਕਟ { ਦੀ ਵਰਤੋਂ ਕੀਤੀ ਹੈ ਜੋ ਕਿ main( ) ਫੰਕਸ਼ਨ ਦੀ ਸ਼ੁਰੂਆਤ ਨੂੰ ਦਰਸ਼ਾਉਂਦਾ ਹੈ।
- ਲਾਈਨ ਨੰ 4 ਅਤੇ 5 ਵਿੱਚ, ਅਸੀਂ ਇੰਟੀਜ਼ਰ ਅਤੇ ਫਲੋਟ ਟਾਈਪ ਦੇ ਵੇਰੀਏਬਲਾਂ int a; ਅਤੇ float pi; ਨੂੰ initialize ਕੀਤਾ ਹੈ ਅਤੇ ਉਹਨਾਂ ਵਿੱਚ ਲਿਟਰਲ (ਸਥਿੱਰ ਮੁੱਲ) ਸਟੋਰ ਕਰਵਾਏ ਹਨ। ਇਹਨਾਂ ਡਿਕਲੇਰੇਸ਼ਨ ਸਟੇਟਮੈਂਟਾਂ ਨੂੰ ਖਾਸ ਚਿੰਨ ਸੈਮੀਕਾਲਨ (;) ਨਾਲ ਖਤਮ (terminate) ਕੀਤਾ ਗਿਆ ਹੈ।
- ਲਾਈਨ ਨੰ 6 ਅਤੇ 7 ਵਿੱਚ, ਅਸੀਂ ਆਉਟਪੁੱਟ ਸਟੇਟਮੈਂਟ printf( ) ਫੰਕਸ਼ਨ ਦੀ ਵਰਤੋਂ ਇੰਟੀਜ਼ਰ ਅਤੇ ਫਲੋਟ ਵੇਰੀਏਬਲਾਂ a ਅਤੇ pi ਦੇ ਮੁੱਲਾਂ ਨੂੰ ਦਰਸਾਉਂਣ ਲਈ ਕੀਤੀ ਹੈ। ਵੇਰੀਏਬਲਾਂ ਦੇ ਮੁੱਲਾਂ ਨੂੰ ਦਰਸਾਉਂਣ ਲਈ ਫਾਰਮੇਟ ਸਟਰਿੰਗ %d ਅਤੇ %f ਦੀ ਵਰਤੋਂ ਕੀਤੀ ਗਈ ਹੈ।
- ਲਾਈਨ ਨੰ 8 ਵਿੱਚ, ਅਸੀਂ ਘੁੰਡੀਦਾਰ ਬਰੈਕਟ } ਦੀ ਵਰਤੋਂ ਕੀਤੀ ਹੈ ਜੋ main ਫੰਕਸ਼ਨ ਦੇ ਅੰਤ ਨੂੰ ਦਰਸ਼ਾਉਂਦਾ ਹੈ।

ਉਪਰੋਕਤ ਪ੍ਰੋਗਰਾਮ ਦੀ ਲਾਈਨ 4 ਅਤੇ 5 ਵਿੱਚ, ਅਸੀਂ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਕੋਡ ਦੀ ਵਿਆਖਿਆ ਕਰਨ ਲਈ ਇੱਕ ਵਿਸੇਸ ਚਿੰਨ੍ਹ // ਦੀ ਵਰਤੋਂ ਕੀਤੀ ਹੈ। ਚਿੰਨ੍ਹ // ਨਾਲ ਸੁਰੂ ਹੋਈਆਂ ਲਾਈਨਾਂ ਨੂੰ ਕਮੈਂਟਸ (Comments) ਕਿਹਾ ਜਾਂਦਾ ਹੈ। ਕਮੈਂਟਸ ਪ੍ਰੋਗਰਾਮਿੰਗ ਵਿੱਚ ਸਾਡੇ ਕੋਡ ਦਾ ਵਰਣਨ ਕਰਨ ਲਈ ਵਰਤੇ ਜਾਂਦੇ ਹਨ। ਇਹਨਾਂ ਨੂੰ ਕੰਪਾਈਲੇਸ਼ਨ ਪ੍ਰਕਿਰਿਆ ਦੇ ਦੌਰਾਨ ਕੰਪਾਈਲਰ ਦੁਆਰਾ ਨਜ਼ਰ-ਅੰਦਾਜ ਕੀਤਾ ਜਾਂਦਾ ਹੈ। ਚਿੰਨ੍ਹ // **ਸਿੰਗਲ ਲਾਈਨ ਕਮੈਂਟ** ਲਈ ਵਰਤਿਆ ਜਾਂਦਾ ਹੈ। ਬਹੁਤੀਆਂ-ਲਾਈਨ ਵਾਲੇ ਕਮੈਂਟ ਲਈ, ਅਸੀਂ ਆਪਣੇ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ /* ਅਤੇ */ ਚਿੰਨ੍ਹਾਂ ਦੀ ਵਰਤੋਂ ਕਰ ਸਕਦੇ ਹਾਂ।

ਪ੍ਰੋਗਰਾਮ 7.4: ਇੰਟੀਜ਼ਰ ਵੇਰੀਏਬਲਾਂ ਨਾਲ ਇਨਪੁੱਟ/ਆਉਟਪੁੱਟ ਸਟੇਟਮੈਂਟਾਂ ਦੀ ਵਰਤੋਂ ਨੂੰ ਦਰਸਾਉਂਦਾ ਸੀ ਭਾਸ਼ਾ ਵਿੱਚ ਪ੍ਰੋਗਰਾਮ।



Program 7.4



Output of Program 7.4

ਪ੍ਰੋਗਰਾਮ 7.5: ਫਲੋਟ ਵੇਰੀਏਬਲਾਂ ਨਾਲ ਇਨਪੁੱਟ/ਆਉਟਪੁੱਟ ਸਟੇਟਮੈਂਟਾ ਦੀ ਵਰਤੋਂ ਨੂੰ ਦਰਸਾਉਂਦਾ ਸੀ-ਪ੍ਰੋਗਰਾਮ

```c
#include<stdio.h>
void main( )
{
float radius;    //float variable declaration
printf("Input Value of radius ");
scanf("%f",&radius);
printf("Value of radius is %f",radius);
}
```

Program: 7.5

```
C:\Users\Kansal\Documents\s1.exe
Input Value of radius 1.25
Value of radius is 1.250000
Process returned 27 (0x1B)   e
Press any key to continue.
```

Output of Program: 7.5

ਪ੍ਰੋਗਰਾਮ 7.6: ਕਾਂਸੈਂਟਟ ਮੁੱਲਾਂ ਦੀ ਵਰਤੋਂ ਨੂੰ ਦਰਸਾਉਂਦਾ ਸੀ-ਪ੍ਰੋਗਰਾਮ

```c
#include<stdio.h>
void main( )
{
const float pi=3.14;     //float constant
printf("Value of pi is %f", pi);
}
```

Program: 7.6

```
C:\Users\Kansal\Documents\s1.exe
Value of pi is 3.140000
Process returned 23 (0x17)
Press any key to continue.
```

Output of Program: 7.6

ਜੇ ਅਸੀਂ ਸੀ ਪ੍ਰੋਗਰਾਮਿੰਗ ਲਈ Turbo C ਦੀ ਵਰਤੋਂ ਕਰਦੇ ਹਾਂ, ਤਾਂ ਪ੍ਰੋਗਰਾਮ ਨੂੰ ਕੰਪਾਈਲ ਕਰਨ ਅਤੇ ਚਲਾਉਣ ਤੋਂ ਬਾਅਦ ਆਉਟਪੁੱਟ-ਵਿੰਡੋ ਸਿੱਧੇ ਤੌਰ ਤੇ ਪ੍ਰਦਰਸ਼ਿਤ ਨਹੀਂ ਹੋਵੇਗੀ। ਆਉਟਪੁੱਟ ਵਿੰਡੋ ਨੂੰ ਵੇਖਣ ਲਈ Alt+F5 ਦਬਾਓ ਜਾਂ Windows ਮੀਨੂ ਨੂੰ ਖੋਲ੍ਹੋ ਅਤੇ ਫਿਰ ਆਉਟਪੁੱਟ ਵਿੰਡੋ ਨੂੰ ਵੇਖਣ ਲਈ User Screen ਤੇ ਕਲਿੱਕ ਕਰੋ। ਪਰ ਜੇ ਅਸੀਂ ਪ੍ਰੋਗਰਾਮਿੰਗ ਲਈ code::blocks ਦੀ ਵਰਤੋਂ ਕਰਦੇ ਹਾਂ, ਤਾਂ ਪ੍ਰੋਗਰਾਮ ਦੇ ਸਫਲਤਾਪੂਰਵਕ ਚੱਲਣ ਤੋਂ ਬਾਅਦ ਆਉਟਪੁੱਟ ਵਿੰਡੋ ਆਪਣੇ ਆਪ ਹੀ ਸਾਡੀ ਸਕ੍ਰੀਨ ਤੇ ਨਜ਼ਰ ਆ ਜਾਵੇਗੀ।

ਯਾਦ ਰੱਖਣ ਯੋਗ ਗੱਲਾਂ

1. ਸੀ ਇੱਕ **ਜਨਰਲ-ਪਰਪਜ਼ ਪ੍ਰੋਗਰਾਮਿੰਗ** ਭਾਸ਼ਾ ਹੈ ਜੋ ਡੈਨਿਸ ਰਿਚੀ ਦੁਆਰਾ ਤਿਆਰ ਕੀਤੀ ਗਈ ਸੀ।
2. ਸੀ ਭਾਸ਼ਾ ਨੂੰ ਵਿਸ਼ੇਸ਼ ਤੌਰ ਤੇ **ਮਿਡਲ ਲੇਵਲ ਭਾਸ਼ਾ** ਦਾ ਦਰਜਾ ਦਿਤਾ ਗਿਆ ਹੈ ਕਿਉਂਕਿ ਇਸ ਵਿੱਚ ਦੋਵੇਂ – ਲੋਅ ਲੇਵਲ ਅਤੇ ਹਾਈ ਲੇਵਲ ਭਾਸ਼ਾਵਾਂ ਦੀਆਂ ਯੋਗਤਾਵਾਂ ਹਨ।
3. IDE ਇੱਕ **ਇੰਟੀਗ੍ਰੇਟਿਡ ਡਿਵੈਲਪਮੈਂਟ ਇਨਵਾਇਰਮੈਂਟ** ਹੈ ਜੋ ਇੱਕ ਅਜਿਹਾ ਵਾਤਾਵਰਣ ਪ੍ਰਦਾਨ ਕਰਦਾ ਹੈ ਜਿਸ ਵਿੱਚ ਪ੍ਰੋਗਰਾਮ ਲਿਖਣ ਤੋਂ ਇਲਾਵਾ ਪ੍ਰੋਗਰਾਮ ਨੂੰ ਕੰਪਾਈਲ ਕਰਨ, ਚਲਾਉਣ, ਟੈਸਟ ਅਤੇ ਡੀਬੱਗ ਆਦਿ ਕਰਨ ਦੀ ਸੁਵਿਧਾ ਵੀ ਹੁੰਦੀ ਹੈ।
4. **ਕਰੈਕਟਰ ਸੈਟ** ਪ੍ਰੋਗਰਾਮ ਬਨਾਉਣ ਵਿੱਚ ਸਾਰੇ ਵਰਤਣਯੋਗ ਕਰੈਕਟਰਾਂ ਦਾ ਸਮੂਹ ਹੁੰਦਾ ਹੈ।
5. **ਟੋਕਨ** ਇੱਕ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਸਭ ਤੋਂ ਛੋਟੀਆਂ ਵਿਅਕਤੀਗਤ ਇਕਾਈਆਂ ਹੁੰਦੀਆਂ ਹਨ। ਇਹ ਅੰਗਰੇਜੀ ਭਾਸਾ ਵਿੱਚ ਸਬਦਾਂ ਅਤੇ ਵਿਰਾਮ ਚਿੰਨ੍ਹਾਂ ਵਰਗੇ ਹੁੰਦੇ ਹਨ।
6. **ਕੀਵਰਡਸ** ਨੂੰ ਰਿਜ਼ਰਵ ਵਰਡਸ (reserve words) ਵੀ ਕਿਹਾ ਜਾਂਦਾ ਹੈ ਜਿਨ੍ਹਾਂ ਦਾ ਅਰਥ ਪਹਿਲਾਂ ਤੋਂ ਹੀ ਪਰਿਭਾਸ਼ਿਤ ਕੀਤਾ ਗਿਆ ਹੁੰਦਾ ਹੈ।
7. **ਆਈਡੈਂਟੀਫਾਇਰਜ਼** ਪ੍ਰੋਗਰਾਮ ਦੇ ਐਲੀਮੈਂਟਸ, ਜਿਵੇਂ ਕਿ ਵੇਰੀਏਬਲ, ਕਾਂਸਟੈਂਟਸ, ਐਰੇ, ਫੰਕਸਨ, ਸਟਰਕਚਰ ਆਦਿ, ਨੂੰ ਦਿੱਤੇ ਨਾਮ ਹਨ।

8. **ਆਪਰੇਟਰ** ਉਹ ਚਿੰਨ੍ਹ ਹੁੰਦੇ ਹਨ ਜੋ ਕਿ ਕੁਝ ਗਣਿਤਕ ਜਾਂ ਲਾਜੀਕਲ ਓਪਰੇਸ਼ਨ ਕਰਨ ਲਈ ਵਰਤੇ ਜਾਂਦੇ ਹਨ।

9. **ਕਾਂਸਟੈਂਟ** ਆਪਣਾ ਮੁੱਲ ਪ੍ਰੋਗਰਾਮ ਚੱਲਣ ਦੌਰਾਨ (during execution) ਬਦਲਣ ਦੀ ਆਗਿਆ ਨਹੀਂ ਦਿੰਦੇ ਜਦੋਂ ਕਿ ਵੇਰੀਏਬਲ ਦਾ ਮੁੱਲ ਬਦਲਿਆ ਜਾ ਸਕਦਾ ਹੈ।

10. **ਡਾਟਾ ਟਾਈਪਸ** ਵੇਰੀਏਬਲ ਜਾਂ ਹੋਰ ਪ੍ਰੋਗਰਾਮ ਐਲੀਮੈਂਟਾਂ ਲਈ ਮੁੱਲਾਂ ਦੀ ਇੱਕ ਖਾਸ ਸੀਮਾ (range) ਨੂੰ ਪਰਿਭਾਸ਼ਿਤ ਕਰਦੀਆਂ ਹਨ।

11. scanf( ) ਅਤੇ printf( ) **ਫਾਰਮੈਟਿਡ ਇਨਪੁੱਟ/ਆਊਟਪੁੱਟ ਫੰਕਸ਼ਨ** ਹਨ ਜਿਨ੍ਹਾਂ ਦੀ ਵਰਤੋਂ ਕਿਸੇ ਵੀ ਕਿਸਮ ਦੇ ਨੁਮੈਰੀਕਲ, ਸਿੰਗਲ ਕਰੈਕਟਰ, ਅਤੇ ਸਟ੍ਰਿੰਗ ਮੁੱਲਾਂ ਦੀ **ਇਨਪੁੱਟ/ਆਊਟਪੁੱਟ** ਕਰਨ ਲਈ ਕੀਤੀ ਜਾਂਦੀ ਹੈ।

12. ਕਮੈਂਟਸ (Comments) ਪ੍ਰੋਗਰਾਮਿੰਗ ਵਿੱਚ ਸਾਡੇ ਕੋਡ ਦਾ ਵਰਣਨ ਕਰਨ ਲਈ ਵਰਤੇ ਜਾਂਦੇ ਹਨ।

ਅਭਿਆਸ

ਭਾਗ-ੳ

ਪ੍ਰ 1. ਬਹੁਪਸੰਦੀ ਪ੍ਰਸ਼ਨ:

I. ਸੀ ਇੱਕ _____________ ਪਰਪਜ਼ ਵਾਲੀ ਪ੍ਰੋਗਰਾਮਿੰਗ ਭਾਸ਼ਾ ਹੈ।

a. special b. general
c. objective d. ਇਹਨਾਂ ਵਿਚੋਂ ਕੋਈ ਨਹੀਂ

II. ਹੇਠ ਲਿਖਿਆਂ ਵਿਚੋਂ ਕਿਹੜੀ ਆਈਡੈਂਟੀਫਾਇਰ ਦੀ ਸਹੀ ਉਦਾਹਰਣ ਨਹੀਂ ਹੈ ?

a. roll_no b. %age_marks
c. rollno d. main

III. ਹੇਠਾਂ ਲਿਖਿਆਂ ਵਿਚੋਂ ਕਿਹੜਾ ਟੋਕਨ ਹੁੰਦਾ ਹੈ ?

a. ਕੀਅਵਰਡ b. ਵਿਸ਼ੇਸ਼ ਚਿੰਨ੍ਹ
c. ਲਿਟਰਲਜ਼ d. ਇਹ ਸਾਰੇ

IV. ਹੇਠ ਲਿਖਿਆਂ ਵਿਚੋਂ ਕਿਹੜਾ ਕੀਵਰਡ ਡਾਟਾ-ਟਾਈਪ ਨੂੰ ਨਹੀਂ ਦਰਸਾਉਂਦਾ ?

a. int b. float
c. const d. char

V. _____________ ਦੀ ਵਰਤੋਂ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਕੋਡ ਦਾ ਵਰਨਣ ਕਰਨ ਲਈ ਕੀਤੀ ਜਾਂਦੀ ਹੈ।

a. ਕੰਪਾਈਲਰ b. ਕਮੈਂਟਸ
c. ਲਿਟਰਲਜ਼ d. ਆਈਡੈਂਟੀਫਾਇਰ

2. ਖਾਲੀ ਥਾਵਾਂ ਭਰੋ:

I. __________ ਇੱਕ ਪ੍ਰੋਗਰਾਮ ਵਿੱਚ ਸਭ ਤੋਂ ਛੋਟੀਆਂ ਵਿਅਕਤੀਗਤ ਇਕਾਈਆਂ ਹੁੰਦੀਆਂ ਹਨ।

II. ਪ੍ਰੋਗਰਾਮ ਦੇ ਐਲੀਮੈਂਟਸ, ਜਿਵੇਂ ਕਿ ਵੇਰੀਏਬਲ, ਕਾਂਸਟੈਂਟਸ, ਐਰੇ, ਫੰਕਸਨ, ਸਟਰਕਚਰ ਆਦਿ, ਨੂੰ ਦਿੱਤੇ ਨਾਮ ਨੂੰ __________ ਕਿਹਾ ਜਾਂਦਾ ਹੈ।

III. ਉਹ ਪ੍ਰੋਗਰਾਮ ਦੇ ਤੱਤ ਜੋ ਆਪਣਾ ਮੁੱਲ ਪ੍ਰੋਗਰਾਮ ਚੱਲਣ ਦੌਰਾਨ ਬਦਲਣ ਦੀ ਆਗਿਆ ਨਹੀਂ ਦਿੰਦੇ, ਨੂੰ __________ ਕਿਹਾ ਜਾਂਦਾ ਹੈ।

IV. ਸਿੰਗਲ ਪ੍ਰਿਸੀਜ਼ਨ (Single Precision) ਮੁੱਲਾਂ ਨਾਲ ਕੰਮ ਕਰਨ ਲਈ ਅਸੀਂ __________ ਡਾਟਾ ਟਾਈਪ ਦੀ ਵਰਤੋਂ ਕਰਦੇ ਹਾਂ।

V. ਹੈਡਰ ਫਾਈਲ ਦੀ ਐਕਸਟੈਂਸ਼ਨ __________ ਹੁੰਦੀ ਹੈ।