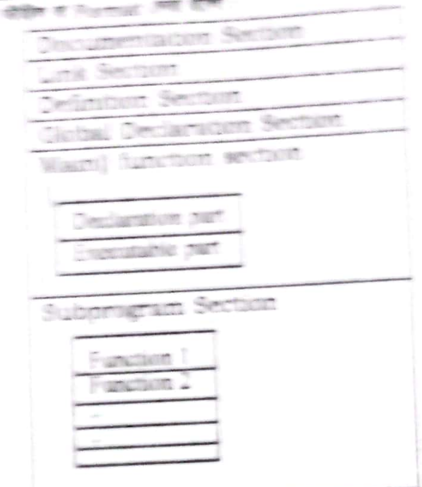
নিচে সি-প্রোগ্রামের প্রোগ্রামিং স্টাইল বা Format দেয়া হলো :



চিত্র-৩.১ : সি-প্রোগ্রামের প্রোগ্রামিং স্টাইল

প্রোগ্রামিং স্টাইল (Programming Style) : প্রোগ্রামিং স্টেটমেন্টসমূহকে যুক্তিগ্রাহ্যভাবে লেখার রীতিকে প্রোগ্রামিং স্টাইল (Programming Style) বলে। C-কে Free form ল্যাংগুয়েজ বলা হয়। এর অর্থ হচ্ছে কোথায় লাইন টাইপ শুরু হবে– কম্পাইলার তা গ্রাহ্য করে না। তবে যত্রতত্র প্রোগ্রাম স্টেটমেন্ট লেখার অভ্যাস পরিহার করা উচিত। তাতে বরং প্রোগ্রামের পাঠযোগ্যতা বাড়ে।

প্রোগ্রাম লেখার অনেক বিশুদ্ধ রীতি রয়েছে। তবে প্রোগ্রামের সাথে সঙ্গতিপূর্ণ একটি স্টাইল বেছে নিলে ভালো হয়। প্রোগ্রামিং স্টাইল বার বার পরিবর্তন না করে একটির মধ্যে সীমাবদ্ধ থাকলে ভিন্ন মাত্রার পরিবেশে প্রোগ্রাম পরিচালনায় বেশি স্বাচ্ছন্দ্য অনুভব করা যায়।

C প্রোগ্রামিং এ Program লিখার জন্য যদিও একাধিক Style প্রচলিত তথাপি যে কোন একটি Style ব্যবহার করে প্রোগ্রাম লিখা খুবই গুরুত্বপূর্ণ বিষয়।

নিচে C Programming Style এর একটি নমুনা দেয়া হলো :

```
# include <stdio.h>
void main ()
{
clrscr ();
printf ("Dhaka Polytechnic Institute");
}
```

সি তে প্রোগ্রাম লিখার ক্ষেত্রে একাধিক Style থাকলেও প্রোগ্রাম কোডিং এর নিম্নবর্ণিত বিষয়সমূহ বিবেচ্য। যথা :

১. প্রতিটি সি প্রোগ্রামই ছোট হাতের অক্ষরে লিখতে হয় ব্যতিক্রম শুধু Symbolic Constant এর বেলায়।

২. সি প্রোগ্রামের বিশেষ কিছু Statement ছাড়া প্রতিটি Statement Semicolon দ্বারা শেষ হয়। সুতরা Statement এর শেষে Semicolon ব্যবহার করে একই লাইনে একাধিক Statement লিখা যায়।

৩. Braces '{ }' বা দ্বিতীয় বন্ধনী দ্বারা ফাংশনের শুরু ও শেষ বুঝায়। একাধিক প্রোগ্রাম লাইনকে একত্রে সমন্বিত কর উদ্দেশ্যেও Brace ব্যবহার করা হয়। যথোপযুক্ত স্থানে এই Brace ব্যবহার করলে প্রোগ্রামটি দেখতে আকর্ষণীয় হ রিডিং ও ডিবাগিং প্রক্রিয়া সহজতর হয়। অন্যথায় প্রোগ্রামে ত্রুটি (Error) পরিলক্ষিত হয়। তাই যথাযথভাবে Brac ব্যবহার করতে হবে।

৪. সাধারণ কিছু ব্যতিক্রম ছাড়া কম্পাইলার ফাঁকা লাইন এবং ফাঁকা স্থানসহ প্রোগ্রাম কোড গণনা করে।

৩.১ সি ভাষায় প্রোগ্রাম লিখার কারণ উল্লেখকরণ

(Mention the Reasons for Writing Program in C)

কোন বিশেষ [illegible] মাইক্রোকন্ট্রোলারের জন্য নির্দিষ্ট অ্যাসেম্বলি ভাষা দিয়ে [illegible]। এই অ্যাসেম্বলি ভাষাতে সাধারণত মেশিন ভাষা (যেমন– 0 × 12 0 × 07 0 × 44 0 × 87) এর পরিবর্তে [illegible] mnemonic (যেমন : MOV 22 MYBUFFER + 7) ব্যবহার হয়। একটি অ্যাসেম্বলার অ্যাসেম্বলি ভাষাকে মেশিন ভাষায় রূপান্তর করে। কিন্তু অ্যাসেম্বলি ভাষা কোন বিশেষ মাইক্রোকন্ট্রোলারের জন্য লিখিত। ফলে মাইক্রোকন্ট্রোলার পরিবর্তন করলে এর জন্য বিশেষ অ্যাসেম্বলি ভাষাও জানতে হয়। পক্ষান্তরে, সি (C) সাধারণ উদ্দেশ্য [illegible] একটি উচ্চতর ভাষা, কম্পাইলার থাকলে এটি যে কোন মাইক্রোকন্ট্রোলারে ব্যবহার হতে পারে। তবে এর একটি অসুবিধা হলো– প্রোগ্রাম বড় হয় এবং ফলে বড় প্রোগ্রাম মেমোরির প্রয়োজন হয়। তাই ছোট কোন মেমোরির মাইক্রোকন্ট্রোলারের ক্ষেত্রে এটি অসুবিধা। নিচে মাইক্রোকন্ট্রোলারে সি প্রোগ্রাম ব্যবহারের সুবিধা উল্লেখ করা হলো :

১। ফাংশন লাইব্রেরির কোড ব্যবহার করা যায়।

২। সি প্রোগ্রাম সহজ বোধগম্য এবং একে সংশোধনও সহজ।

৩। অ্যাসেম্বলি ভাষার তুলনায় সি ভাষায় প্রোগ্রাম রচনা সহজ এবং সময় অনেক কম লাগে।

৪। একে পরিবর্তন ও হালনাগাদ করা সহজ।

৫। সব ধরনের মাইক্রোকন্ট্রোলারে প্রয়োগ করা যায়। ফলে ভিন্ন ভিন্ন মাইক্রোকন্ট্রোলারের জন্য ভিন্ন ভিন্ন এসেম্বলি ভাষা জানা প্রয়োজন নেই।

৬। এটা সুবহ (portable) অর্থাৎ একে সামান্য পরিবর্তন করে বা কোনরূপ পরিবর্তন না করেই অন্য মাইক্রোকন্ট্রোলারে ব্যবহার করা যায়।

৩.২ 8051 এর জন্য সি ডাটার ধরন ও অপারেটরের তালিকা

(List C Data Types and Operators for 8051)

8051 এর জন্য সি ডাটার ধরন (C data types for 8051) :

8051 এর জন্য প্রোগ্রামিং "সি" এর ডাটার ধরনের উপর ভাল ধারণা প্রোগ্রামারকে ছোট hex file তৈরি করতে সহায়তা করে। নিচে সি প্রোগ্রামে বহুল ব্যবহৃত ডাটা শ্রেণির আকার ও রেঞ্জের তালিকা দেওয়া হলো :

ডাটার শ্রেণি	আকার (bit)	ডাটা রেঞ্জ/ব্যবহার
Unsigned char	8-bit	0 – 255
(Signed) char	8-bit	(– 128) – (+ 127)
Unsigned int	16-bit	0 – 65535
(signed) int	16-bit	(– 32768) – (+ 32767)
Sbit (single bit)	1-bit	শুধুমাত্র SFR bit-addressable
Bit	1-bit	RAM bit-addressable
Sfr	8-bit	শুধুমাত্র RAM addressable 80-FFH

8051 এর জন্য সি অপারেটর (C operator for 8051) :

প্রোগ্রামিং "সি" এর রাশিমালায় ব্যবহৃত অপারেটরের তালিকা নিচে উল্লেখ করা হলো :

(ক) Data access and size operator

অপারেটর	নাম	উদাহরণ	বর্ণনা
[ ]	Array element	x[6]	অ্যারে x এর ৭ম উপাদান
.	Member selection	portb.2	পোর্ট D এর বিট 2
->	Member selection	pstruct->x	pstruct দ্বারা structure এর মেম্বার নির্দেশিত হয়।
*	Indirection	*p	অ্যাড্রেস p এ মেমোরি স্থানের ধারক।
&	Address of	&x	চলক x এর অ্যাড্রেস।

(খ) Relational and logical operator

অপারেটর	নাম	উদাহরণ	বর্ণনা
>	Greater than	x > y	যদি x এর মান y এর চেয়ে বড় হয়, তবে লজিক 1, অন্যথায় 0 হয়।
>=	Greater than or equal to	x > = y	যদি x এর মান y এর চেয়ে বড় বা সমান হয়, তবে লজিক 1, অন্যথায় 0 হয়।
<	Less than	x < y	যদি x এর মান y এর চেয়ে ছোট হয়, তবে লজিক 1, অন্যথায় 0 হয়।
< =	Less than or equal	x < = y	যদি x এর মান y এর চেয়ে ছোট হয় বা সমান হয়, তবে লজিক 1, অন্যথায় 0 হয়।
=	Equal to	x = y	যদি x এর মান y এর সমান হয়, তবে লজিক 1, অন্যথায় 0 হয়।
!=	Not equal to	x != y	যদি x এর মান y মান সমান না হয়, তবে লজিক 1, অন্যথায় 0 হয়।
!	Logical NOT	!x	যদি x এর মান 0 হয়, তবে লজিক 1, অন্যথায় 0 হয়।
&&	Logical AND	x&&y	যদি x বা y এর মান 0 হয়, তবে লজিক 0, অন্যথায় 1 হয়।
||	Logical OR	x||y	যদি x ও y এর মান উভয়ই 0 হয়, তবে লজিক 0, অন্যথায় 1 হয়।

(গ) Arithmetic operator

অপারেটর	নাম	উদাহরণ	বর্ণনা
*	Multiplication	x*y	x কে y দ্বারা গুণ করে।
/	Division	x/y	x কে y দ্বারা ভাগ করে।
%	Modulo	x%y	x কে y দ্বারা ভাগ করার পর ভাগশেষ নির্ণয় করে।
+	Addition	x + y	x এর সাথে y এর যোগ করে।
-	Subtraction	x - y	x থেকে y এর বিয়োগ করে।
++	Increment	x ++	এটি ব্যবহারের পর x এর 1 বৃদ্ধি করে।
- -	Decrement	- - x	এটি ব্যবহারের পূর্বে x এর মান 1 হ্রাস করে।
+	Unary Plus	+ x	x-কে ধনাত্মক প্রদর্শন করে।
-	Negation	- x	x কে - 1 দ্বারা গুণ করে।

(ঘ) Bitwise operator

অপারেটর	নাম	উদাহরণ	বর্ণনা
~	Bitwise complement NOT	~ x	বিট 1 কে 0 তে এবং বিট 0 কে 1 এ রূপান্তর করে।
&	Bitwise AND	x&y	x ও y এর Bitwise AND
|	Bitwise OR	x|y	x ও y এর Bitwise OR
^	Bitwise Exclusive OR	x ^ y	x ও y এর XOR
<<	Left shift	x<<2	x এর বিট বাম দিকে 2 বিট পজিশনে স্থানান্তরিত হয়।
>>	Right shift	x >> 2	x এর বিট ডান দিকে 2 বিট পজিশনে স্থানান্তরিত হয়।

(ঙ) Miscellaneous operator

অপারেটর	নাম	উদাহরণ	বর্ণনা
( )	Function	wait (10)	10 এর একটি আর্গুমেন্টের সাথে অপেক্ষা।
(type)	Type cast	(double)x	x একটি double-এ রূপান্তরিত হয়।
?	Conditional	x?y z	যদি x শূন্য না হয়, তবে y মূল্যায়ন করে, অন্যথায় z মূল্যায়ন করে।
	Sequential evaluation	x ++, y++	প্রথমে x এর ক্রমবৃদ্ধি হয় এবং পরে y এর ক্রমবৃদ্ধি হয়।

৩.৩ 8051 এর জন্য সি ভাষায় টাইম ডিলে তৈরির প্রক্রিয়া বর্ণনা
(Describe Creating Time Delay in C)

8051 মাইক্রোকন্ট্রোলারের জন্য সি ভাষায় দুই ভাবে টাইম ডিলে করা যায় যথা :

১। একটি সরল লুপ ব্যবহার করে।

২। 8051 টাইমার ব্যবহার করে।

উভয়ক্ষেত্রেই ডিলে একটি অসিলোস্কোপে বা একটি সিমুলেটর ব্যবহার করে দেখা যায়। এখানে একটি সরল লুপ ব্যবহার করে কীভাবে টাইম ডিলে তৈরি করা হয় তা আলোচনা করা হলো। লুপ ব্যবহার করে টাইম ডিলে তৈরির জন্য তিনটি ফ্যাক্টর বিবেচনা করতে হয়, যা ডিলে এর সঠিকতায় প্রভাব বিস্তার করতে পারে :

১। মেশিন সাইকেলের সংখ্যা এবং প্রতি মেশিন সাইকেলে ক্লকের সংখ্যা।

২। 8051 এর সিস্টেম ফ্রিকুয়েন্সি। এটি XTAL1 ও XTAL2 এর মধ্যে সংযুক্ত ক্রিস্টালের ফ্রিকুয়েন্সি। মেশিন সাইকেলের জন্য ক্লক পিরিয়ডের সময়কাল ক্রিস্টাল ফ্রিকুয়েন্সির একটি ফাংশন।

৩। 8051 এর সি কম্পাইলার বাছাই।

অ্যাসেম্বলি ভাষা প্রোগ্রামিং এ যেহেতু নির্দেশনা সংখ্যা এবং প্রতি নির্দেশনায় সাইকেল সংখ্যা জানা, তাই উপযুক্ত ডিলে জানা ও নিয়ন্ত্রণ করা যায়। সি প্রোগ্রামের ক্ষেত্রে সি কম্পাইলার সি স্টেটমেন্টসমূহকে অনুবাদ করে এবং অ্যাসেম্বলি ভাষা প্রোগ্রামে ফাংশনে রূপান্তর করে। তাই ভিন্ন ভিন্ন কম্পাইলার ভিন্ন ভিন্ন কোড উৎপন্ন করে। নিচে লুপ ব্যবহার করে 8051 এর P1 পোর্টের বিটগুলোকে সামান্য ডিলেসহ toggle করার জন্য একটি সি প্রোগ্রাম দেখানো হলো।

```
#include < reg51.h>
Void main (void)
    {
        Unsighned int x;
        For (;;)
            {
                P1 = 0x55;
                For (x =0; x < 4000; x ++);
                P1 = 0 x AA;
                For (x = 0 x AA;
                For (x = 0; x < 4000; x ++)
            }
    }
```

৩.৪ 8051 এর জন্য সি প্রোগ্রাম
(Write Program in C for Sending Data to Port, Accessing Code ROM, Data Serialization and Interrupt Operation)

প্রোগ্রাম- ১ : 8051 এর পোর্ট P1 এ - 4 থেকে + 4 মানগুলো পাঠানোর জন্য সি প্রোগ্রাম লিখ।

(Write an 8051 C program to send values of - 4 to + 4 port P1)

সমাধান :

```
// Signed numbers
# include <reg51.h>
void main (void)
{
char mynum [] = { + 1, - 1, + 2, - 2, + 3, - 3, + 4, - 4};
unsigned char z;
for (z = 0; z < = 8, z ++)
pl = mynum [z];
}
```

প্রোগ্রাম- ২ : 8051 এর পোর্ট P1 এ মান পাঠানোর জন্য সি প্রোগ্রাম লিখ।

(Write a 8051 C program to send values 00-FF to port P1)

সমাধান :

```
# include <reg51.h>
void main (void)
{
unsigned char z;
for (z = 0; z < = 255; z ++)
P1 = z;
}
```

প্রোগ্রাম- ৩ : 8051 এর পোর্ট P1 এর মাধ্যমে সিরিয়ালি এক বিট করে 44H মানকে প্রেরণের জন্য সি প্রোগ্রাম লিখ। LSB (Least Significant Bit) প্রথমে যেতে হবে। (Write a C program to send out the value 44H serially one bit at a time via P1.0. The LSB should go out first)

সমাধান :

```
# include < reg51.h>
sbit p1b0 = p1^0;
sbit regALSB = ACC^0;
void main (void)
{
unsigned char conbyte = 0 × 44;
unsigned char z;
ACC = conbyte;
for (z = 0; z < 8; z ++)
    {
    P1b0 = regALSB;
    ACC = ACC >>1;
    }
}
```

প্রোগ্রাম-৪ : 8051 এর পোর্ট P1 এ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C ও D এর ASCII অক্ষর পাঠানোর জন্য সি প্রোগ্রাম লিখ।

(Write an 8051 C program to send hex values for ASCII characters of 0, 1, 2, 3, 4, 5 A, B, C, and D to port P1)

সমাধান :

```
#include <reg51.h>
void main (void)
{
unsigned char mynum [ ] = "012345ABCD";
unsigned char z;
for (z = 0; z < = 10; z ++)
p1 = mynum [z];
}
```

প্রোগ্রাম-৫ : 8051 এর পোর্ট P1 এর মাধ্যমে সিরিয়ালি এক বিট করে 44H মানকে আনয়নের সি প্রোগ্রাম লিখ। LSB (Least Significant Bit) প্রথমে আনয়ন করতে হবে।

(Write a C program to bring in a byte of data serially one bit at a time via P1.0. The LSB should come in first)

সমাধান :

```
# include < reg51.h>
sbit P1b0 = P1^0;
sbit ACCMSB = ACC^7;
bit membit;
void main (void)
{
unsigned char x;
for (x = 0; x < 8; x ++)
    {
    membit = P1b0;
    ACC = ACC >>1;
    ACCMSB = membit;
    }
P2 = ACC;
}
```

প্রোগ্রাম- ৬ : 8051 এর পোর্ট P1 এর মাধ্যমে সিরিয়ালি এক বিট করে 44H মানকে আনার জন্য সি প্রোগ্রাম লিখ। MSB (Most Significant Bit) প্রথমে আনয়ন করতে হবে। (Write a C program to bring in a byte of data serially one bit at a time via P1.0. The MSB should come in first)

সমাধান :

```
# include < reg51.h>
sbit p1b0 = P1^0;
sbit regALSB = ACC^0;
bit membit;
void main (void)
{
unsigned char x;
for (x = 0; x < 8; x ++)
    {
    membit = P1b0;
    ACC = ACC <<1;
    regALSB = membit;
    }
P2 = ACC;
}
```