স্টেটমেন্ট

লেকচার-১৮

স্টেটমেন্ট

লেকচার-১৮

এই পাঠ শেষে যা যা শিখতে পারবে-

১। continue স্টেটমেন্ট ব্যবহার করে প্রোগ্রাম লিখতে পারবে।

২। break স্টেটমেন্ট ব্যবহার করে প্রোগ্রাম লিখতে পারবে।

৩। goto স্টেটমেন্ট ব্যবহার করে প্রোগ্রাম লিখতে পারবে।

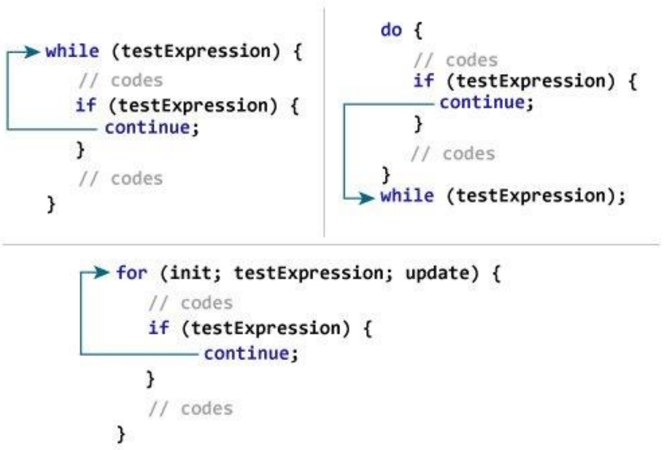
continue স্টেটমেন্ট

‘সি’ প্রোগ্রামে লুপ কন্ট্রোল স্টেটমেন্টের লুপ বডির এক বা একধিক স্টেটমেন্ট নির্বাহ না হয়ে পুনরায় প্রথম থেকে নির্বাহের জন্য continue স্টেটমেন্ট ব্যবহৃত হয়। continue স্টেটমেন্ট শর্তযুক্ত এবং শর্তবিহীন উভয় ভাবে ব্যবহার করা যায়। তবে শর্তবিহীন continue স্টেটমেন্ট অসীম লুপের সৃষ্টি করে।

continue স্টেটমেন্টের ফরম্যাট

continue;

continue স্টেটমেন্ট যেভাবে কাজ করে-



```
while (testExpression) {
    // codes
    if (testExpression) {
      continue;
    }
    // codes
}
```

```
do {
   // codes
   if (testExpression) {
     continue;
   }
   // codes
}
while (testExpression);
```

```
for (init; testExpression; update) {
    // codes
    if (testExpression) {
       continue;
    }
    // codes
}
```

for লুপে continue স্টেটমেন্ট ব্যবহার করে ১ থেকে ১০ এর মধ্যে অবস্থিত বিজোড় সংখ্যাগুলো দেখানোর প্রোগ্রাম।

```
/* for loop ব্যবহার করে প্রোগ্রাম */
#include<stdio.h>
#include<conio.h>
main()
{
     int i;
     for(i=1; i<=10; i=i+1)
     {
          if(i%2==0)
             continue;
          printf("%d\t ",i);
     }
     getch();
}
```

উপরের প্রোগ্রামে লুপ বডির Test Expression যখন সত্য হয় তখন continue স্টেটমেন্টটি কাজ করে। অর্থাৎ printf() ফাংশনটি নির্বাহ না করে প্রোগ্রামের নির্বাহ আবার লুপের প্রথম থেকে শুরু হয়।

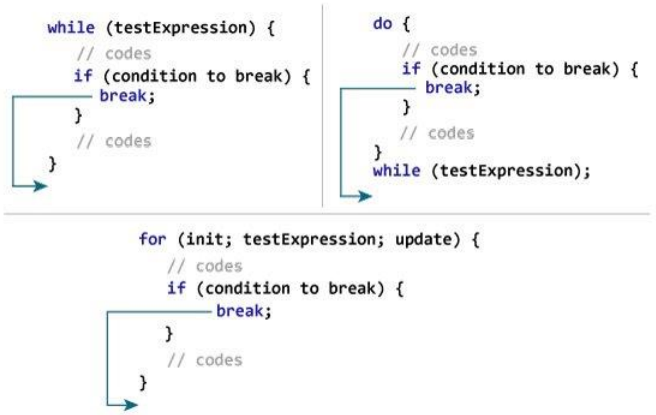
break স্টেটমেন্ট

লুপ কন্ট্রোল স্টেটমেন্টের লুপ বডির স্টেটমেন্টগুলো সাধারণত Test Expression মিথ্যা না হওয়া পর্যন্ত পুনরাবৃত্তি করতে থাকে। কিন্তু Test Expression মিথ্যা হওয়ার পুর্বেই লুপ থেকে বের হওয়ার জন্য break স্টেটমেন্ট ব্যবহার করা হয়। break স্টেটমেন্ট loops অথবা switch স্টেটমেন্টে ব্যবহৃত হয়। যখন break স্টেটমেন্ট কাজ করে তখন প্রোগ্রাম কন্ট্রোল লুপ থেকে বের হয়ে যায় এবং লুপের বাইরে প্রথম স্টেটমেন্ট থেকে প্রোগ্রাম নির্বাহ হতে থাকে। নেস্টেড লুপের ক্ষেত্রে প্রোগ্রাম কন্ট্রোল, প্রথমে ভেতরের লুপ থেকে বের হয়ে আসে এবং পরে বাইরের লুপ থেকে বের হয়ে আসে। সাধারণত break স্টেটমেন্ট এমন একটি অবস্থায় ব্যবহার করা হয় যখন লুপটি কতবার পুনরাবৃত্তি হবে তা আমাদের কাছে অজানা অথবা কোন শর্তের ভিত্তিতে প্রোগ্রাম কন্ট্রোল লুপ থেকে বের হয়ে আসা।

break স্টেটমেন্টের ফরম্যাট

break;

break স্টেটমেন্ট যেভাবে কাজ করে-



```
while (testExpression) {
   // codes
   if (condition to break) {
      break;
   }
   // codes
}
```

```
do {
   // codes
   if (condition to break) {
      break;
   }
   // codes
}
while (testExpression);
```

```
for (init; testExpression; update) {
   // codes
   if (condition to break) {
      break;
   }
   // codes
}
```

নিচের প্রোগ্রামটি লক্ষ কর-

```
/* for loop ব্যবহার করে প্রোগ্রাম */
#include<stdio.h>
#include<conio.h>
main()
{
    int i;
    for(i=1; i<=5; i=i+1)
    {
        printf("Bangladesh\n");
        if(i==3)
           break;
    }
    getch();
}
```

উপরের প্রোগ্রামের লুপটি পাঁচ বার পুনরাবৃত্তি হওয়ার পরিবর্তে তিনবার পুনরাবৃত্তি হবে। কারণ যখন লুপ বডির কন্ডিশনটি সত্য হবে তখন break স্টেটমেন্টটি

কাজ করবে অর্থাৎ প্রোগ্রাম কন্ট্রোল লুপ থেকে বের হয়ে যাবে। ফলে Bangladesh লেখাটি তিনবার দেখাবে।

goto স্টেটমেন্ট

goto স্টেটমেন্টকে জাম্পিং স্টেটমেন্ট বলা হয়। ‘সি’ ভাষায় প্রোগ্রাম নির্বাহের নিয়ন্ত্রণ শর্তযুক্ত বা শর্তবিহীন ভাবে এক স্টেটমেন্ট থেকে উপরে বা নিচে অপর কোন স্টেটমেন্টে বা প্রোগ্রামের পূর্বনির্ধারিত কোন স্থানে স্থানান্তরের জন্য goto স্টেটমেন্ট ব্যবহার করা হয়। একটি নির্দিস্ট শর্তের জন্য প্রোগ্রামের একটি নির্দিস্ট অংশ পুনরাবৃত্তি করতেও goto স্টেটমেন্ট ব্যবহার করা যায়। এছাড়া মাল্টিপল লুপ ব্রেক করতেও goto স্টেটমেন্ট ব্যবহার করা যায় যা একটি সিঙ্গেল break স্টেটমেন্ট দিয়ে সম্ভব নয়। যাইহোক, goto স্টেটমেন্টের এর ব্যবহার খুবই কম। কারণ এটি প্রোগ্রামকে জটিল ও বিভ্রান্তি করে।

goto স্টেটমেন্টের ফরম্যাট

```
label:
----------
----------
goto label;
```

অথবা

```
goto label;
-----------
-----------
label:
```

এখানে label প্রোগ্রামারের দেওয়া একটি আইডেন্টিফায়ার, যেখানে প্রোগ্রাম নির্বাহের নিয়ন্ত্রন স্থানান্তরিত হবে। এই আইডেন্টিফায়ার লেখার জন্য আইডেন্টিফায়ার লেখার নিয়ম মেনে লিখতে হবে। মনে রাখতে হবে, আইডেন্টিফায়ার (label) এর পরে সেমিকোলন(;) না হয়ে কোলন(:) হয়।

দুটি পূর্ণ সংখ্যার ল.সা.গু. নির্নয়ের জন্য ‘সি’ প্রোগ্রামিং ভাষায় লেখা নিচের প্রোগ্রামটি লক্ষ করি।

```
#include<stdio.h>
#include<conio.h>
main()
{
     int a,b,l;
     printf("Enter the two numbers: ");
     scanf("%d %d",&a,&b);
     if(a>b)
        l=a;
     else
        l=b;
     again:
     if(l%a==0 && l%b==0)
        printf("LCM of %d and %d is %d",a,b,l);
     else
     {
        l=l+1;
        goto again;
     }
    getch();
}
```

উপরের প্রোগ্রামটিতে goto স্টেটমেন্ট ব্যবহার করে প্রোগ্রাম নির্বাহের নিয়ন্ত্রণ উপরে নির্দিস্ট জায়গায় স্থানান্তর করা হয়েছে।