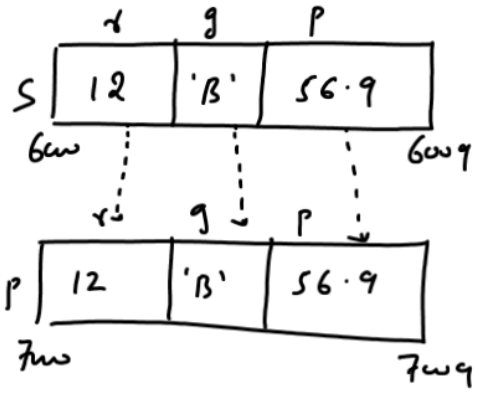
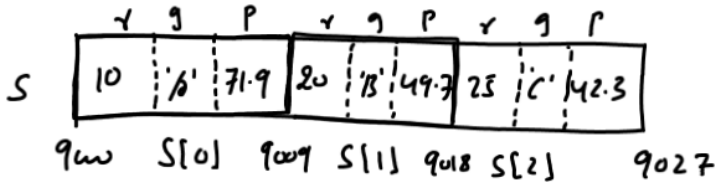
Copying One Structure Variable To Another




```
struct Student
{
   int roll;
   char grade;
   float per;
};
int main()
{
   struct Student S,P;
   printf("Enter roll,grade and per:");
   scanf("%d %c %f",&S.roll,&S.grade,&S.per);
   P.roll=S.roll;
   P.grade=S.grade;   OR   P = S;
   P.per=S.per;
   printf("Roll=%d,Grade=%c,Per=%f",P.roll,P.grade,P.per);
  return 0;
}
```

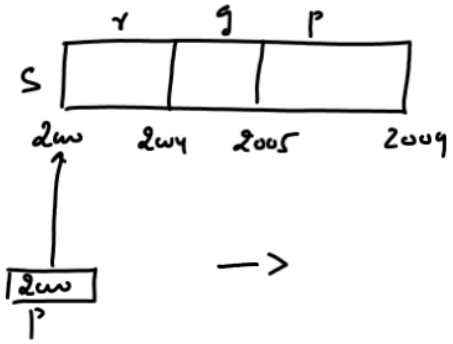
```
   struct Student
   {
      int roll;
      char grade;
      float per;
   };
   int main()
   {
      struct Student S,P;
      printf("Enter roll,grade and per:");
      scanf("%d %c %f",&S.roll,&S.grade,&S.per);
      P.roll=S.roll;
      P.grade=S.grade;
      P.per=S.per;
      printf("Roll=%d,Grade=%c,Per=%f",P.roll,P.grade,P.per);
     return 0;
   }
```

CREATING ARRAY OF STRUCTURE




```
struct Student
{
   int roll;
   char grade;
   float per;
};
int main()
{
   struct Student S[3];
   int i;
   for(i=0;i<3;i++)
   {
     printf("Enter roll,grade and per:");
     scanf("%d %c %f",&S[i].roll,&S[i].grade,&S[i].per);
   }
   for(i=0;i<3;i++)
      printf("\n%d %c %f",S[i].roll,S[i].grade,S[i].per);
return 0;
}
```

STRUCTURE AND POINTER




```
struct Student
{
   int roll;
   char grade;
   float per;
};
int main()
{
   struct Student S,*P;
   P=&S;
   P->roll=10;
   P->grade='A';
   P->per=71.5;
   printf("Roll=%d,Grade=%c,Per=%f",P->roll,P->grade,P->per);
   return 0;
}
```

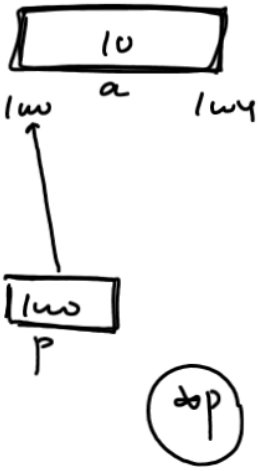
S.roll=10;

P.roll=10; ✗

a.) (*P).roll=10; ✓

S.roll=10;

b) P→roll=10;

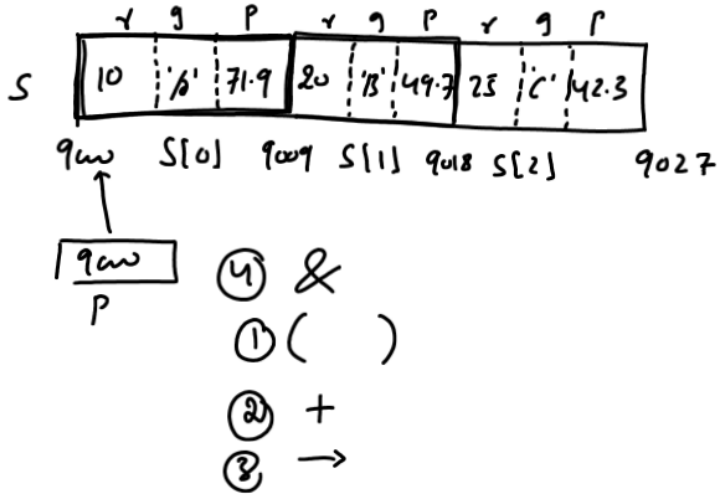



```
int a = 10;
int *p;
p = &a;
*p = 15;
a = 15;
```
Same

ACCEPTING INPUT FROM USER IN STRUCTURE USING POINTER

```
struct Student
{
   int roll;
   char grade;
   float per;
};
int main()
{
   struct Student S,*P;
   P=&S;
   printf("Enter roll,grade and per:");
   scanf("%d %c %f",&P->roll,&P->grade,&P->per);
   printf("\nRoll=%d,Grade=%c,Per=%f",P->roll,P->grade,P->per);
   return 0;
}
```

ACCESSING ARRAY OF STRUCTURE USING POINTER



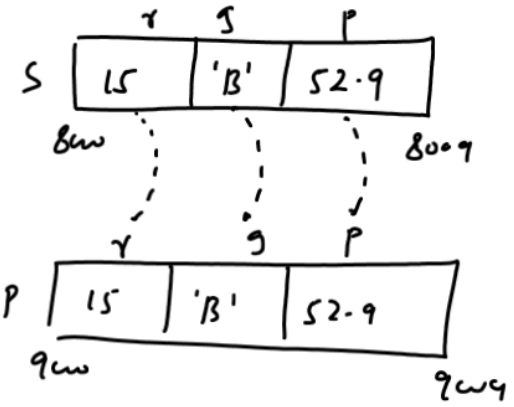

```
struct Student
{
  int roll;
  char grade;
  float per;
};
int main()
{
  struct Student S[3],*P;
  int i;
  P=S;
  for(i=0;i<3;i++)
  {
    printf("Enter roll,grade and per:");
    scanf("%d %c %f",&(P+i)->roll,&(P+i)->grade,&(P+i)->per);
  }
  for(i=0;i<3;i++)
     printf("\n%d %c %f",(P+i)->roll,(P+i)->grade,(P+i)->per);
return 0;
}
```

P+i => P + i * sizeof(data type of P)

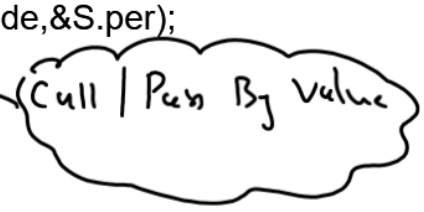
9000 + 1 * 9

```
struct Student
{
   int roll;
   char grade;
   float per;
};
int main()
{
   struct Student S[3],*P;
   int i;
   P=S;
   for(i=0;i<3;i++)
   {
     printf("Enter roll,grade and per:");
     scanf("%d %c %f",&(P+i)->roll,&(P+i)->grade,&(P+i)->per);
   }
   for(i=0;i<3;i++)
      printf("\n%d %c %f",(P+i)->roll,(P+i)->grade,(P+i)->per);
return 0;
}
```

STRUCTURE AND FUNCTION




```
struct Student
{
   int roll;
   char grade;
   float per;
};
void display(struct Student);
int main()
{
   struct Student S;
   printf("Enter roll,grade and per:");
   scanf("%d %c %f",&S.roll,&S.grade,&S.per);
   display(S);
   return 0;
}
void display(struct Student P)
{
   printf("Roll=%d,Grade=%c,Per=%f",P.roll,P.grade,P.per);
}
```



```
struct Student
{
   int roll;
   char grade;
   float per;
};
void display(struct Student);
void accept(struct Student *);
int main()
{
   struct Student S;
   accept(&S);
   display(S);
   return 0;
}
void accept(struct Student *P)
{
  printf("Enter roll,grade and per:");
  scanf("%d %c %f",&P->roll,&P->grade,&P->per);
}
```

```
void display(struct Student P)
{
   printf("Roll=%d,Grade=%c,Per=%f",P.roll,P.grade,P.per)
}
```